HABILITATION A DIRIGER DES RECHERCHES
Université de Bretagne-Sud
membre de l'Université européenne de Bretagne
Ecole doctorale SICMA

présentée par
André Rossi
Préparée au Lab-STICC (UMR 6285)
Laboratoire des Sciences et Techniques de l'Information, de la Communication et de la Connaissance

De l'optimisation dans les réseaux

Habilitation à diriger des recherches soutenue le 14 septembre 2012
devant le jury composé de :

Alexandre Dolgui
Professeur à l'Ecole Supérieure des Mines de St Etienne / *rapporteur*
Bernard Penz
Professeur à l'INP Grenoble / *rapporteur*
Daniel Vanderpooten
Professeur à l'Université Paris-Dauphine / *rapporteur*

Christian Artigues
Directeur de recherche au LAAS (Toulouse) / *examinateur et président du jury*
Xavier Gandibleux
Professeur à l'Université de Nantes / *examinateur*
Marc Sevaux
Professeur à l'Université de Bretagne-Sud / *examinateur*

Résumé

Ce mémoire d'habilitation à diriger des recherches traite de problèmes d'optimisation dans les réseaux, dont la modélisation et la résolution reposent sur les outils de la recherche opérationnelle. Le premier chapitre retrace brièvement mon parcours de formation, ainsi que mes activités d'enseignement et de recherche. Le second chapitre est consacré aux réseaux de capteurs sans fil. La minimisation du défaut de couverture, la maximisation de la durée de vie et l'exploration de compromis entre ces deux objectifs sont abordés à l'aide d'un algorithme de génération de colonnes combiné à un algorithme génétique. Le troisième chapitre traite de la configuration robuste d'un réseau de distribution d'électricité visant à le prémunir des effets néfastes d'une augmentation de la demande. Un algorithme basé sur la génération de plans coupants associé à des inégalités valides est notamment proposé, puis comparé avec une formulation basée sur la programmation linéaire en nombres entiers et une heuristique. Le quatrième chapitre aborde le problème du déploiement (ou la mise à niveau) d'un réseau de fibre optique permettant la communication multicast, avec pour but de minimiser le coût des équipements opto-électroniques à installer. Deux versions du problème, qui se distinguent par l'expression du coût des équipements en question, sont abordés à l'aide d'un algorithme de plans coupants enrichi par une fonction de réparation et une recherche tabou, illustrant l'efficacité d'une étroite collaboration entre méthodes exactes et approchées. Enfin, ce mémoire est clos par un chapitre consacré à la présentation de perspectives ouvertes par ces travaux, ainsi qu'une réflexion plus personnelle sur l'exercice de la direction de recherche au niveau individuel, dans l'encadrement doctoral, et dans la direction d'équipe.

Summary

This *habilitation à diriger des recherches* addresses network optimizations problems, whose modeling and solution processes are based on operations research. The first chapter begins with a summary of my education, and synthesizes my research and teaching activities. The second chapter is dedicated to wireless sensor networks. Breach covering minimization, lifetime maximization and trade offs between these two conflicting objectives are addressed with a column generation algorithm hybridized with a genetic algorithm. The third chapter deals with the robust configuration of an electricity distribution network, for preventing the nasty effects of demand increase on the network operation. A cutting plane algorithm along with a class of valid inequalities are proposed, then compared with a mixed integer linear programming formulation and a greedy heuristic. The fourth chapter introduces two fiber optic network deployment problems that aim at minimizing the cost of opto-electronic equipment for enabling multicast communication. The two problems, that differ by their objective function, are both addressed with a cutting plane algorithm hybridized with a repair function and a tabu search procedure that illustrate the beneficial effect of collaboration between exact and approximation algorithms. Finally, this thesis closes with a chapter devoted to future works and with a personal reflection on research direction on the individual level, in the context of doctoral supervision, and for research team leadership.

Avant-propos

Je tiens tout d'abord à remercier messieurs Alexandre Dolgui, Bernard Penz et Daniel Vanderpooten qui me font l'honneur d'être rapporteurs de ce mémoire d'habilitation à diriger des recherches. Je remercie également messieurs Christian Artigues, Xavier Gandibleux et Marc Sevaux d'avoir accepté de prendre part au jury en tant qu'examinateurs.

Les travaux présentés dans ce mémoire doivent beaucoup aux nombreux séjours que j'ai eu la chance d'effectuer à l'Université d'Hyderabad, et à la collaboration particulièrement fructueuse avec Alok Singh, *reader* dans cette université. Qu'il trouve ici l'expression de ma gratitude et de mon amitié, notamment pour m'avoir permis de découvrir - et d'aimer - l'Inde. Au laboratoire (LESTER, puis Lab-STICC), Marc Sevaux a également joué un rôle déterminant pour mon évolution professionnelle en me permettant de progresser en recherche opérationnelle, et en me faisant profiter de son réseau et de ses conseils avisés.

Je remercie les anciens doctorants que j'ai eu le privilège de co-encadrer (Aïssam et María), et ceux que je co-encadre aujourd'hui (Boureima, Alain et Fabian), chacun d'eux m'a permis de progresser au plan scientifique comme au plan de l'encadrement doctoral. C'est à Pascal Berruet, Jean-Luc Philippe et Marc Sevaux que je suis redevable de ces précieux co-encadrements de thèse, je les en remercie vivement.

Depuis mon arrivée à Lorient, je mesure combien il fait bon vivre et travailler au Lab-STICC. Je voudrais en remercier ici tous les membres, à commencer par son directeur, monsieur Guy Gogniat, ainsi que Florence et Virginie, dont la bonne humeur et le professionnalisme sont au cœur du labo. J'exprime également toute ma gratitude aux collègues de l'ENSIBS, et en particulier à son directeur Jean-Luc Philippe et à sa directrice des études Jeanne Villaneau, qui ont rendu possibles mes séjours indiens en aménageant mon emploi du temps.

Je n'oublie pas l'accueil chaleureux du personnel technique et administratif de l'UBS, ni les nombreuses fois où j'ai pu compter sur son aide. Amélie, Annaïg, Céline, Christophe, Isabelle, Jocelyne, Luc, Mireille, Patrick, Philippe et Thierry ont toute mon amitié.

André Rossi
Mai 2012

Table des matières

Chapitre 1

Introduction générale

Introduction

Ce mémoire d'habilitation à diriger des recherches dresse un bilan de mes activités post-doctorales, où mes activités de recherche sont plus particulièrement détaillées. Plutôt que de tenter de produire une synthèse uniforme ou chronologique de mes travaux scientifiques, j'ai choisi de développer ici mes contributions à la fois les plus récentes et les plus représentatives de mon orientation thématique.

La présente **introduction générale** est divisée en quatre sections. La première est un curriculum vitæ étendu, la seconde dresse une vue d'ensemble de mes activités de recherche ; elle est suivie de la liste de mes publications post-doctorales. La quatrième partie fait état de mes activités d'enseignement. Cette introduction générale est suivie de trois chapitres présentant mes contributions scientifiques récentes de manière plus détaillée.

Le chapitre 2 est consacré à l'optimisation de la qualité de service et de la durée de vie des réseaux de capteurs sans fil. Ce travail a débuté en 2009 lors de mon premier séjour à l'université d'Hyderabad. Il montre comment l'utilisation d'un algorithme génétique peut accélérer la convergence d'un algorithme de génération de colonnes, et propose quelques pistes pour explorer les compromis entre qualité de service et durée de vie. Ce chapitre est une adaptation d'un article à paraître dans Networks [60], co-écrit avec Alok Singh et Marc Sevaux.

Le chapitre 3 aborde le problème de la configuration d'un réseau de distribution d'électricité basse tension confronté à l'augmentation de la demande énergétique. Cette augmentation met en péril son équilibre, ce qui peut se traduire par un black out, c'est-à-dire un effondrement du réseau électrique à très grande échelle. L'approche proposée se fonde sur un algorithme de génération de plans coupants, accompagné de quelques résultats théoriques sur la complexité et la non-garantie de performance d'une heuristique. Ce chapitre est une adaptation d'un travail datant de 2010 et publié dans EJOR [58], en collaboration avec Alexis Aubry et Mireille Jacomino.

Le chapitre 4 étudie deux problèmes proches relevant de la communication multicast dans les réseaux de fibre optique. L'objectif est de minimiser le coût des appareils opto-électroniques à ajouter à un réseau de fibre optique existant (ou à concevoir), sous deux hypothèses relatives au coût desdits équipements. Comme dans le chapitre 2, on propose une méthode exacte (cette fois-ci un algorithme de plans coupants) dont on améliore sensiblement les résultats à l'aide d'une métaheuristique (une recherche tabou en l'occurrence). Ce chapitre est une adaptation d'un travail commencé en 2011 à l'université d'Hyderabad avec Alok Singh et Shyam Sundar. Une première approche de ce problème, purement métaheuristique, a été publiée dans Information Sciences [64].

Mes travaux les plus récents, développés à l'université d'Hyderabad en 2012 et consacrés au problème de l'arbre dominant de poids minimum, ne me semblent pas avoir atteint un degré de maturité suffisant pour faire l'objet d'un chapitre. Ils se situent cependant dans la lignée du chapitre 4, et ont donné lieu à l'implémentation d'un algorithme de branch-and-cut tirant parti de métaheuristiques.

Le chapitre 5 tire des conclusions et expose quelques perspectives ouvertes par les travaux présentés, et se termine par quelques réflexions personnelles sur la direction de recherche.

Une annexe suit les références bibliographiques, elle est constituée des trois articles suivants :

- Un article à paraître dans *Computers and Operations Research* co-écrit avec Alok Singh et Marc Sevaux, qui traite de l'optimisation de la durée de vie des réseaux de capteurs sans fil dont le rayon de couverture est ajustable, lorsque l'ajustement est continu et lorsqu'il est discret. Ce travail peut être vu comme une extension du chapitre 2.
- Un article publié dans *Annals of Operations Research* fin 2011 co-écrit avec Alexis Aubry et Mireille Jacomino, qui propose une analyse de sensibilité relativement détaillée sur un problème d'ordonnancement préemptif de machines identiques partiellement multifonctions. Il offre une analyse fine de la dégradation de performances (ici la date de fin de traitement de la commande) en fonction de l'augmentation de la demande de chaque famille de produits. Ce travail témoigne de mon intérêt pour les problèmes de robustesse et d'aide à la décision, et l'illustre sur un problème différent de celui du chapitre 3.
- Un article publié dans *Discrete Applied Mathematics* en 2011 co-écrit avec María Soto et Marc Sevaux sur l'établissement de nouvelles bornes supérieures pour le problème de coloration des sommets d'un graphe. Ce travail réalisé pendant la thèse de María Soto est une contribution plus théorique qui a découlé de l'étude du problème de l'allocation des structures de données en mémoire dans les systèmes embarqués.

1.1 Curriculum vitæ

1.1.1 Données personnelles

Nom : Rossi
Prénom : André
Date de naissance : 3 mai 1976 à Avignon (Vaucluse)
Nationalité : Française

Adresse professionnelle :
Lab-STICC (UMR CNRS 6285)
Centre de recherche
Université de Bretagne-Sud
BP 92116
F-56321 Lorient Cedex

Téléphone : +033 2 97 87 45 25
Adresse électronique : `andre.rossi@univ-ubs.fr`
Page personnelle : `http://www-labsticc.univ-ubs.fr/~rossi/`

Adresse personnelle :
139 rue Paul Guieysse
56100 Lorient

1.1.2 Expérience professionnelle

Depuis 2005 Maître de conférences à l'Université de Bretagne-Sud, Lorient (Lab-STICC et ENSIBS).

2004–2005 ATER à temps complet à l'Université de Bretagne-Sud, Lorient.

2003–2004 ATER à temps complet à l'Ecole Nationale Supérieure d'Ingénieurs Electriciens de Grenoble (ENSIEG), école de l'Institut National Polytechnique de Grenoble (INPG), Grenoble. Cette école s'appelle aujourd'hui ENSE3.

2000–2003 Doctorant au Laboratoire d'Automatique de Grenoble (LAG, INPG), sous la direction de Mireille Jacomino. Moniteur à l'Ecole Supérieure d'Ingénieurs en Systèmes Industriels Avancés de Rhône-Alpes, à Valence (ESISAR, INPG).

1.1.3 Formation

2003 Doctorat de l'INPG, *Ordonnancement en milieu incertain, mise en œuvre d'une démarche robuste*, sous la direction de Mireille Jacomino. Ce travail a reçu le *Prix de thèse* de l'INPG en 2005.

2000 DEA d'automatique productique au LAG (INPG), mention bien.

2000 Diplôme d'ingénieur de l'ENSIEG (INPG), option productique, mention bien.

1.1.4 Encadrement de thèses soutenues

- Co-encadrement de María Soto, avec Marc Sevaux, *Graph coloring, Application to Memory Allocation in Electronic Chip Design* (2008–2011). Madame Soto est actuellement ATER à l'UBS, à Vannes.
- Co-encadrement d'Aïssam Belabbas, avec Jean-Luc Philippe, *On the safe autonomous wheelchair navigation in a health care center* (2004–2007). Monsieur Belabbas occupe depuis un poste d'ingénieur à la société Sydel (Lorient).

1.1.5 Encadrement de thèses en cours

- Co-encadrement de Fabian Castaño avec Marc Sevaux et Nubia Velasco, *Global approach for sensor networks* (2012–2015).
- Co-encadrement de Boureima Zerbo avec Jean-Charles Créput et Marc Sevaux, *On the routing of packets in a QoS-constrained Network-on-Chip environment* (2008–2012).

- Co-encadrement d'Alain Bignon avec Pascal Berruet, *Model driven engineering for the joint generation of control and human-machine interfaces for the fuel distribution system in large ships* (2009–2012). Thèse CIFRE avec la société Segula.

Par ailleurs, j'ai été impliqué de manière informelle dans l'encadrement doctoral de Florent De Lamotte, Jean-Louis Lallican et Kods Trabelsi (j'ai pris part à leur jury de thèse).

1.1.6 Participation à des jurys de thèse

- Dr. Evgeny Gurevsky (École des Mines de St Etienne, 2011)
- Dr. María Soto (UBS, 2011)
- Dr. Kods Trabelsi (UBS, 2009)
- Dr. Aissam Belabbas (UBS, 2007)
- Dr. Jean-Louis Lallican (UBS, 2007)
- Dr. Florent de Lamotte (UBS, 2006)
- Dr. Duy Long Ha (INPG, 2006)

1.1.7 Autres activités relevant de l'encadrement doctoral

- Encadrement d'étudiants de Master recherche
  Mohamed Abdul Basith Ameer Abdul Kader, Master of Science, University of Massachusetts, 2010, Amherst, USA (en collaboration avec Maciej Ciesielski).
  Abderrahim Falih, Master EII, Université de Bretagne-Sud, 2008, est maintenant enseignant à l'IGA (Maroc).
  Karim Braikia, Master ASP, Université de Nantes, 2007, a poursuivi en thèse au LATTIS (France).
  Soumia Kessal, Master ASP, Université de Nantes, 2007, a poursuivi en thèse à l'ENST (France).

- Stage de recherche
  Jai Raj Bhattacharya, IIIT Hyderabad, Stage de recherche en 2009. Cet étudiant a poursuivi ses études à IIIT Hyderabad (Inde).

1.1.8 Activités scientifiques internationales

- Professeur invité au Department of Computer and Information Sciences, de l'Université d'Hyderabad (Inde) pour 2 mois en 2009, 2010, 2011 et 2012.
- Orateur invité par la section IEEE d'Hyderabad (Inde) en février 2012.
  `https://meetings.vtools.ieee.org/meeting_view/list_meeting/10703`
- Trois séminaires en Chine en 2008, en tant que membre de la délégation de l'UBS à Central University of Beijing, Changsha University et Shandong University (à Weihai).
- Quatre séminaires en Inde en 2008 et 2009 en tant que membre de la délégation de l'UBS à University of Madras, University of Hyderabad, MERI à Delhi et IIT Roorkee.

1.1.9 Activités scientifiques nationales

- Bénéficiaire de la Prime d'Excellence Scientifique depuis 2010.
- Récipiendaire en 2005 du Prix de thèse de l'INPG.
- Membre de la ROADEF, du GOThA, membre du GDR-RO (*Groupe Robustesse* animé par Bernard Roy et *Problématiques d'optimisation discrète en micro-électronique*).
- Orateur invité par le *Groupe Robustesse* à l'Université Paris Dauphine en 2007 et 2008.
- Orateur invité aux séminaires du Lab-STICC pour promouvoir la modélisation mathématique pour l'électronique.

1.1.10 Projets

- Projet régional PRIR PSES (Pilotage des Systèmes Embarqués Sûrs) avec les sociétés DCNS et Sydel en 2006–2008. Ce projet visait à appliquer les techniques développées pour les systèmes reconfigurables à un système de conditionnement (Sydel) et à l'appareil à gouverner de sous-marins (DCNS).

- Projet ANR AFANA `http://recherche.telecom-bretagne.eu/afana/fr/` Ce projet a permis, sur la période 2007–2010, de démarrer la thèse en co-tutelle de Boureima Zerbo, et de proposer les premiers modèles visant à représenter le trafic des paquets dans les réseaux sur puce.
- Projet ANR FAON (l'adresse électronique est trop longue pour apparaître ici, rechercher « ANR FAON » avec un moteur de recherche). Ce projet, qui a démarré en 2011 avec le CEA et France Telecom, est dédié à l'étude de la technologie des réseaux large bande optique passifs. Il comporte une partie consacrée à l'allocation de fréquences dans ce réseau, afin d'allouer au mieux la bande passante aux utilisateurs.
- Mission de consultance auprès de Bolloré Plastic Films Division en 2009 (mise au point d'un programme de cutting stock pour la découpe de film plastique). Un projet impliquant l'UBS est en cours de conclusion avec le même partenaire industriel pour l'année 2012.

1.1.11 Implication dans la communauté scientifique

- Arbitre pour Journal of Global Optimization, International Journal of Production Research, JIMS, ORIJ, INFOR, AJOR, Les annales du LAMSADE, IFAC 2008, INCOM 2009, ISCO 2010 (International Symposium on Combinatorial Optimization), ROADEF 2010, MIWAI 2011.
- Co-organisateur de la session *Design and Control of Manufacturing Systems under Uncertainty* à INCOM 2009 à Moscou.
- Organisateur de la session *Genetic algorithms* dans le stream *Metaheuristics* à EURO 2009 à Bonn.
- Animateur à la Fête de la Science à l'UBS.
- Organisateur de la session *Robustness concerns and multiple criteria decision aid* dans le stream *Multicriteria Decision Aiding I* à EURO 2010 à Lisbonne.
- Membre du comité de programme pour MIWAI 2011 (`http://khamreang.msu.ac.th/miwai11/`), MIWAI 2012 (`http://khamreang.msu.ac.th/miwai12/committees.html`), ICORES 2012 et ICORES 2013 (`http://www.icores.org/`).

1.1.12 Responsabilités administratives

- Membre élu au Conseil Scientifique de l'Université de Bretagne-Sud depuis 2012.
- Responsable du Master CGSEIP-Production en 2008-2009. Cette formation a fermé en 2009 pour devenir la spécialité Génie Industriel de l'ENSIBS.
- Responsable du Master GPP (formation continue) depuis 2008.
- Membre de comités de sélection dans les établissements suivants :
  - Université Paris Dauphine en 2009 (poste MCF 110), 2010 (poste MCF 167), 2011 (poste MCF 204) et 2012 (postes MCF 0258 et MCF 0071)
  - Université de Nantes en 2011 (poste MCF 1038)
  - Université de Bretagne-Sud en 2012 (poste MCF 043)
- Membre du jury des épreuves de TIPE (Concours Communs Polytechniques) depuis 2011.

1.2 Activités de recherche

La section précédente révèle le caractère multi-disciplinaire et mobile de mon parcours de chercheur : j'ai préparé une thèse en ordonnancement dans un laboratoire d'automatique grenoblois, je fais aujourd'hui de l'optimisation dans un laboratoire d'électronique lorientais, et j'ai été invité à séjourner plusieurs mois dans un département d'informatique indien ces quatre dernières années. Il n'est donc pas surprenant que mes activités de recherche reflètent cette diversité de thèmes. Je veux croire que cette diversité est également à l'image de la recherche opérationnelle, qui s'est développée au carrefour de l'informatique, des mathématiques, et de l'économie, et englobe les aspects théoriques, applicatifs et computationnels des problèmes auxquels elle s'intéresse.

On trouvera dans cette section une présentation sommaire des différents thèmes de recherche que j'ai abordés, leurs enjeux, les équipes et les publications afférentes. Ces dernières sont listées dans la section 1.3, et sont référencées avec la convention suivante :

- JI désigne une publication dans un journal international
- FJ désigne une publication dans un journal francophone
- CO désigne une publication dans un chapitre d'ouvrage
- BU désigne une publication dans un bulletin
- CI désigne une publication dans une conférence internationale
- CF désigne une publication dans une conférence francophone

1.2.1 Robustesse et aide à la décision

Collaborateurs : Alexis Aubry (Université Henri Poincaré, Nancy), Olga Battaïa (Ecole Supérieure des Mines de Saint-Etienne), Alexandre Dolgui (Ecole Supérieure des Mines de Saint-Etienne), Marie-Laure Espinouse (Grenoble INP), Evgeny Gurevsky (Université d'Avignon et des Pays de Vaucluse), Mireille Jacomino (Grenoble INP).
Publications : [JI03, JI07, JI09, JI12, JI13, JF02, CO01, CO02, CI09, BU01, CI11, CI19, CF09, CF11] et un article en cours de rédaction.

Il s'agit de mon thème de recherche le plus ancien, puisqu'il remonte à ma thèse. Un contrat avec la société ST Microelectronics en 2001 a constitué le point de départ de plusieurs études sur les machines parallèles partiellement multifonctions. L'objectif est d'ordonnancer les opérations sur ces machines de manière à ce que l'incertitude sur la durée des opérations porte aussi peu préjudice que possible à la date d'achèvement de l'ensemble de ces opérations. Outre ce besoin de robustesse, on s'est intéressé à l'analyse de sensibilité, permettant de connaître la dégradation des performances en fonction de l'amplitude des incertitudes affectant la durée des opérations. En 2010, nous avons abordé le problème de la distribution d'énergie électrique avec cette orientation « robustesse », qui fait l'objet du chapitre 3. Plus récemment, à l'occasion de la soutenance de thèse d'Evgeny Gurevsky en décembre 2011, nous avons commencé un travail visant à produire des solutions maximisant la robustesse (au sens du rayon de stabilité de Sotskov) dans le cas de l'affectation des opérations aux postes de travail constituant une ligne d'assemblage.

1.2.2 Synthèse de haut-niveau

Collaborateurs : Philippe Coussy (UBS), Marc Sevaux (UBS), Kods Trabelsi (UBS).
Publications : [JI10, CI08, CI22, CF08, CF12].

La synthèse de haut niveau désigne l'ensemble des outils de conception assistée par ordinateur qui permettent aux concepteurs de circuits de travailler à « haut niveau d'abstraction », c'est-à-dire sans se soucier de la conception détaillée (jusqu'aux transistors). L'objectif des travaux est de transformer les spécifications fonctionnelles d'une application en un circuit sur FPGA (*field-programmable gate array*, un circuit programmable) de manière à minimiser le nombre et le coût des opérateurs (opérateurs arithmétiques et multiplexeurs en particulier). Les approches proposées reposent essentiellement sur un estimateur capable d'évaluer les besoins en multiplexage d'une architecture donnée, et sur des métaheuristiques visant à minimiser le coût de cette architecture et des multiplexeurs qui lui sont associés. Les bons résultats obtenus par ces méthodes ont permis leur intégration dans le logiciel GAUT (Générateur Automatique d'Unités de Traitement), vitrine du LESTER puis du Lab-STICC depuis une quinzaine d'années.

1.2.3 Affectation de mémoire pour les systèmes embarqués

Collaborateurs : Johann Laurent (UBS), Marc Sevaux (UBS), María Soto (UBS).
Publications : [JI06, JI08, CI03, CI10, CI18, CI21, CF01, CF04, CF05, CF06,] deux articles en cours de rédaction.

L'extension des fonctionnalités des smartphones et autres systèmes électroniques embarqués rend le problème de l'autonomie et de la consommation énergétique de plus en plus critique. Or la complexité des applications à concevoir ou à porter sur ces plateformes est telle qu'il n'est plus possible d'opérer « manuellement » l'affectation des données en mémoire, et c'est un compilateur qui se charge de ce travail, avec des résultats souvent très mauvais en termes de consommation. On s'intéresse ici à un processeur de la firme Texas Instrument qui a la possibilité d'accéder à plusieurs de ses bancs mémoire simultanément, et on propose des méthodes exactes et approchées pour déterminer le placement des données en mémoire conduisant à minimiser le temps d'exécution et la consommation. Le problème, d'abord étudié dans sa version statique et sous des hypothèses simplificatrices, peut être vu comme une combinaison du *bin packing* et du *vertex coloring problem*. Des versions dynamiques plus réalistes sont désormais à l'étude.

1.2.4 Optimisation du trafic des messages dans un réseau sur puce

Collaborateurs : Jean-Charles Créput (UTBM), Rachid Dafali (UBS), Jean-Philippe Diguet (CNRS), Marc Sevaux (UBS), Boureima Zerbo (UBS et Université de Ouagadougou, Burkina Faso).
Publications : [CI20], un article soumis et un autre en cours de rédaction.

La miniaturisation des systèmes embarqués, qui va de paire avec la multiplication des fonctions qu'ils doivent remplir, a conduit à l'émergence de réseaux miniatures reliant les différentes parties d'une même puce. Ces réseaux sur puce se distinguent des réseaux informatiques par les contraintes de bande passante et de temps réel très fortes auxquels ils sont soumis, l'interdiction de perdre des paquets (par collision) et l'impossibilité de stocker les paquets au niveau des nœuds de ces réseaux. Ce travail vise à proposer des solutions au routage de plus en plus complexe dans ces réseaux en minimisant la longueur des chemins empruntés afin de minimiser aussi la longueur de l'entête du message (qui porte l'information sur le chemin à suivre). Une seconde version du problème considère le cas plus difficile où la destination des paquets émis par les nœuds peut changer. Il convient alors de construire une solution capable de supporter ces changements sans qu'aucun conflit d'accès aux arcs du réseau n'apparaisse.

1.2.5 Systèmes socio-techniques

Collaborateurs : Aïssam Belabbas (UBS), Pascal Berruet (UBS), Alain Bignon (UBS et Segula), Florent De Lamotte (UBS), Jean-Louis Lallican (UBS et Sydel), Jean-Luc Philippe (UBS).
Publications : [JF01, CI04, CI12, CI14, CI15, CI16, CF07, CF10] et un article soumis.

Un système socio-techninque est un système physique (électronique, mécanique etc) dont les interactions avec l'homme sont très fréquentes ou critiques pour la sécurité des utilisateurs. Après la thèse de doctorat d'Aïssam Belabbas consacrée au pilotage de fauteuils roulants autonomes en environnement domotisé (calcul d'itinéraires alternatifs en cas de dysfonctionnement des issues), la thèse d'Alain Bignon considère le processus de conception de systèmes socio-techniques plus complexes (comme un navire). Il s'agit cette fois de minimiser les risques d'erreur lors de la conception impliquant des experts d'horizons différents (mécanique, hydraulique, électronique, ergonomie etc) ne partageant pas toujours le même vocabulaire. L'approche proposée repose sur une bibliothèque de composants respectant les usages métiers, et capable de produire automatiquement le code de commande de bas niveau ainsi qu'une ébauche du système d'interaction homme-machine.

1.2.6 Méthode algébrique de vérification des circuits

Collaborateurs : Tariq Ahmad, Mohamed Basith, Maciej Ciesielski, (tous membres de l'université du Massachusetts, Amherst, Etats-Unis).
Publications : [CI01].

La conception de circuits numériques complexes pose le problème de la vérification du comportement attendu. Or pour des raisons évidentes qui tiennent à l'explosion combinatoire, il n'est pas possible de tester exhaustivement tous les vecteurs d'entrée. Nous avons proposé une méthode originale basée sur une représentation des portes OU, ET, des semi-additionneurs (half-adders) et des additionneurs (full-adders) afin de remplir cet objectif. Le circuit à vérifier est transformé en un jeu d'équations linéaires qui sont combinées par un solver (en l'occurrence GLPK) afin de prouver qu'il existe une combinaison linéaire de ces équations satisfaisant la relation algébrique (appelée signature) qui décrit le comportement attendu. La méthode proposée a montré que certaines classes de circuits (en particulier les multiplieurs signés) pouvaient être vérifiées plus efficacement que par les méthodes existantes. La méthode a aussi montré sa capacité à retrouver la signature qu'un circuit inconnu.

1.2.7 Optimisation pour les réseaux de capteurs sans fil

Collaborateurs : Fabian Castaño (UBS et Université des Andes, Colombie), Marc Sevaux (UBS), Alok Singh (Université d'Hyderabad, Inde), Nubia Velasco (Université des Andes, Colombie).
Publications : [JI01, JI02, JI04, CI02, CF03], un article en révision et deux articles en cours de rédaction.

Ce travail a commencé en 2009 lors de mon premier séjour à l'université d'Hyderabad. Les premiers résultats obtenus sont présentés en détail dans le chapitre 2. La contribution principale est la combinaison d'un algorithme génétique et d'un programme linéaire en nombres entiers pour accélérer un algorithme de génération de colonnes. Depuis, cette méthode a été adaptée au cas où un nombre minimal de capteurs actifs est requis pour couvrir une cible, avec des contraintes de connexité sur la constitution des couvertures. Il a aussi été étendu aux réseaux constitués de capteurs dont le rayon de couverture (et donc la consommation énergétique) est ajustable. Plus récemment, Fabian Castaño a débuté une thèse en co-tutelle avec l'université de Bretagne-Sud et l'université des Andes afin d'améliorer ces approches en leur incorporant des procédures de stabilisation et de diversification.

1.2.8 Matheuristiques pour les réseaux de télécommunication

Collaborateurs : Alok Singh (Université d'Hyderabad, Inde), Shyam Sundar (Université de technologie de Nanyang, République de Singapour).
Publications : [JI05, CI05, CI17, CF02] un article en révision et un autre en cours de rédaction.

Le déploiement à moindre coût de réseaux de télécommunication, qu'ils soient filaires ou sans fil pose de très nombreux problèmes d'optimisation. Cette grande variété est dûe aux contraintes qui dépendent très fortement de la technologie utilisée. Les travaux menés dans cet axe se restreignent aux problèmes visant à déterminer un arbre particulier sur un graphe non orienté donné. Après avoir traité des problèmes centrés sur les sommets-branche (MBV et MDS, présentés en détail dans le chapitre 4), on s'oriente vers le problème de l'arbre de poids minimum qui se pose lors de l'exploitation des réseaux de communication sans fil. Les méthodes de résolution proposées sont des approches exactes basées sur la génération de plans coupants et le branch-and-cut, qui exploitent des métaheuristiques pour accroître leur efficacité.

1.3 Liste de publications

1.3.1 Journaux internationaux

JI01 A. Singh, A. Rossi, M. Sevaux, *Matheuristic approaches for Q-coverage problem versions in wireless sensor networks*, accepté dans **Engineering Optimization**.

JI02 A. Rossi, A. Singh, M. Sevaux, *An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges*, accepté dans **Computers & Operations Research**.
doi :10.1016/j.cor.2012.04.001.

JI03 A. Aubry, M. Jacomino, A. Rossi, M-L. Espinouse, *Maximising the configuration robustness for parallel multi-purpose machines under setup cost constraints*, accepté dans **Journal of Scheduling**.
doi :10.1007/s10951-011-0257-6.

JI04 A. Rossi, A. Singh, M. Sevaux, *A column generation algorithm for sensor coverage scheduling under bandwidth constraints*, accepté dans **Networks**.
doi :10.1002/net.

JI05 S. Sundar, A. Singh, A. Rossi, *New heuristics for two bounded-degree spanning tree problems*, **Information Sciences**, 195 (1), pp. 226–240, 2012.
doi :10.1016/j.ins.2012.01.037.

JI06 M. Soto, A. Rossi, M. Sevaux, *A mathematical model and a metaheuristic approach for a memory allocation problem*, **Journal of Heuristics**, 18 (1), pp. 149–167, 2012.
doi :10.1007/s10732-011-9165-3.

JI07 A. Rossi, A. Aubry, M. Jacomino, *Connectivity-and-hop-constrained design of electricity distribution networks*, **European Journal of Operational Research**, 218 (1), pp. 48–57, 2012.
doi :10.1016/j.ejor.2011.10.006.

JI08 M. Soto, A. Rossi, M. Sevaux, *Three new upper bounds on the chromatic number*, **Discrete Applied Mathematics**, 159 (18), pp. 2281–2289, 2011.
doi :10.1016/j.dam.2011.08.005.

JI09 A. Rossi, A. Aubry, M. Jacomino, *A sensitivity analysis to assess the completion time deviation for multi-purpose machines facing demand uncertainty*. **Annals of Operations Research**, 191 (1), pp. 219–249, 2011.
doi :10.1007/s10479-011-0996-y.

JI10 M. Sevaux, A. Singh, A. Rossi, *Tabu search for multiprocessor scheduling : application to high level synthesis*. **Asia-Pacific Journal of Operational Research**, 28 (2), pp. 201–212, 2011.
doi :10.1142/S0217595911003132.

JI11 A. Rossi, A. Singh, M. Sevaux, *A metaheuristic for the fixed job scheduling problem under spread time constraints*. **Computers & Operations Research**, 37 (6), pp. 1045–1054, 2010.
doi :10.1016/j.cor.2009.09.007.

JI12 A. Rossi, *A robustness measure of the configuration of multi-purpose machines*. **International Journal of Production Research**, 48 (4) pp. 1013–1033, 2010.
doi :10.1080/00207540802473997.

JI13 A. Aubry, A. Rossi, M-L. Espinouse, M. Jacomino, *Minimizing setup costs for parallel multi-purpose machines under load-balancing constraints*. **European Journal of Operational Research**, 187 (3), pp. 1115–1125, 2008.
doi :10.1016/j.ejor.2006.05.050.

1.3.2 Journaux francophones

JF01 J-L. Lallican, P. Berruet, A. Rossi, J-L Philippe, *SimSED : un environnement pour modéliser et simuler des systèmes transitiques*. **Journal Européen des Systèmes Automatisés**, 41 (5), pp. 541–566, 2007.
doi :10.3166/jesa.41.541-566.

JF02 A. Rossi, M. Jacomino, M-L. Espinouse, *Étude de robustesse : configuration d'un parc de machines partiellement multifonctions*. **Journal Européen des Systèmes Automatisés**, 38 (3), pp. 373–395, 2004.
doi :10.3166/jesa.38.373-395.

1.3.3 Chapitres d'ouvrages

CO01 M-L. Espinouse, A. Rossi, M. Jacomino, *Flexibility and Robustness in Scheduling*, Chapter 3 *On the Robustness of Multi-Purpose Machines Shop Configuration*. ISTE, Paris, 2007. ISBN 9781905209750, 352 pages.

CO02 M-L. Espinouse, A. Rossi, M. Jacomino, *Flexibilité et Robustesse en ordonnacement*, chapitre Robustesse de la configuration d'un parc de machines partiellement multifonctions, pp. 51–69. Traité IC2, série Informatique et systèmes d'information. Hermès, Paris, 2005. ISBN 2-7462-1028-2, 344 pages.

1.3.4 Bulletins

BU01 A. Rossi, *Obstacles and Avenues to Promoting the Use of Multi Criteria Decision Aiding*, Newsletter of the European Working Group Multicriteria Aid for Decisions, Series 3 (21) , Printemps 2010. pp. 5–7.

1.3.5 Conférences internationales avec comité de lecture (depuis 2006)

CI01 M. Basith, T. Ahmad, A. Rossi, M. Ciesielski, *Algebraic Approach to Arithmetic Design Verification*, Proceedings of Formal Methods in Computer Aided Design (FMCAD 2011) 30 octobre – 2 novembre, Austin (Texas), États-Unis, pp. 67–71, novembre 2011. (Taux d'acceptation : 36.1%).

CI02 A. Rossi, M. Sevaux, A. Singh, M. Geiger, *On the cover scheduling problem in wireless sensor networks*, Lecture Notes in Computer Science (LNCS 6701), Network Optimization : International Network Optimization Conference, Springer, pp. 657–668. ISSN 0302-9743, ISBN 978-3-642-21526-1. INOC 2011, Hamburg, Allemagne, 13–16 juin 2011. (Taux d'acceptation : 57.4%). doi :10.1007/978-3-642-21527-8_73.

CI03 M. Soto, A. Rossi, M. Sevaux, *Two Iterative Metaheuristic Approaches to Dynamic Memory Allocation for Embedded Systems*, Lecture Notes in Computer Science 6622, Evolutionary Computation in Combinatorial Optimization, Springer, pp. 251–261. ISSN 0302-9743, ISBN 978-3-642-20363-3. EvoCOP 2011, Turin, Italie, 27–29 avril 2011. (Taux d'acceptation : 52.4%). doi :10.1007/978-3-642-20364-0_22.

CI04 A. Bignon, P. Berruet, A. Rossi, *Joint generation of controls and interfaces for sociotechnical and reconfigurable systems*, IEEE International Conference on Systems, Man, and Cybernetics (SMC 2010), Istanbul, Turquie, 10–13 octobre 2010, pp. 749–755.

CI05 S. Sundar, A. Singh, A. Rossi, *An artificial bee colony algorithm for the 0-1 multidimensional knapsack problem*, in Springer CCIS ISSN : 1865-0929 Proceedings of the Third International Conference on Contemporary Computing (IC3'2010), Noida, Inde, 9–11 août 2010.

CI06 E. Senn, D. Monnereau, A. Rossi, N. Julien, *Using Integer Linear Programming in Test-Bench Generation for Evaluating Communication Processors*, 12th Euromicro Conference on Digital System Design (DSD2009), Patras, Grèce, 27–29 août 2009.

CI07 A. Singh, M. Sevaux, A. Rossi, *A hybrid grouping genetic algorithm for multiprocessor scheduling*, In Proceedings of the second International Conference on Contemporary Computing, IC3'2009, Noida, Inde, 17–19 août 2009, pp. 1–7. Published by Springer, ISBN 978-3-642-03546-3.

CI08 P. Coussy, A. Rossi, M. Sevaux, K. Sörensen, K. Trabelsi, *VNS for high-level synthesis*, In Proceedings of 8th Metaheuristics International Conference, MIC 2009, Hamburg, Allemagne, 13–16 juillet 2009, pp. 173 :1–173 :10.

CI09 A. Aubry, A. Rossi, M. Jacomino, *A generic off-line approach for dealing with uncertainty in production systems optimisation*, Preprints of the 13th IFAC Symposium on Information Control Problems in Manufacturing (INCOM 2009), Moscou, Russie, 3–5 juin 2009, pp. 1464–1469.

CI10 M. Soto, A. Rossi, M. Sevaux, *Two upper bounds on the chromatic number*, Proceedings of the 8th Cologne-Twente Workshop on Graphs and Combinatorial Optimization, Paris, France, 2–4 juin 2009, pp. 191–194. Disponible sous `http://www.lix.polytechnique.fr/ctw09/ctw09-proceedings.pdf#page=203`

CI11 A. Aubry, A. Rossi, M. Jacomino, *Sensitivity Analysis for the Configuration of a Multi-Purpose Machines Workshop*, Proceedings of the 17th World Congress The International Federation of Automatic Control, Séoul, Corée, 6–11 juillet 2008, pp. 15831–15836. doi :10.3182/20080706-5-KR-1001.2879.

CI12 F. Frizon de Lamotte, P. Berruet, A. Rossi, J.-L. Philippe, *Control Code Generation using Model Engineering for an Electric Train*, Proceedings of the 17th World Congress The International Federation of Automatic Control, Séoul, Corée, 6–11 juillet 2008, pp. 8327–8332. doi :10.3182/20080706-5-KR-1001.1786.

CI13 A. Rossi, M. Sevaux, *Mixed-integer linear programming formulation for High Level Synthesis*, Proceedings of the Eleventh International Workshop on Project Management and Scheduling, Istanbul, Turquie, 28–30 avril 2008, pp. 222–226. ISBN : 978-9944-62-618.

CI14 P.Berruet, J-L. Lallican, A. Rossi, J-L Philippe, *Generation of control for conveying systems based on component approach*, IEEE SMC 2007, Montréal, octobre 2007. doi :10.1109/ICSMC.2007.4413766.

CI15 J-L. Lallican, P. Berruet, A. Rossi, J-L. Philippe, *A component-based approach for conveying system control design*, IFAC ICINCO, Angers, 9–12 mai 2007, pp. 329–336.

CI16 A. Belabbas, P. Berruet, A. Rossi, J-L. Philippe, *A cooperative model generation for disabled people assistance*, Tele-info'06, Istanbul, Turkey, 27–29 mai 2006.

1.3.6 Conférences internationales sans comité de lecture (depuis 2006)

CI17 A. Rossi, *On the dominating tree problem*, EURO XXV, 25th European Conference on Operational Research, Vilnius, Lituanie, 8–11 juillet 2012.

CI18 M. Soto, A. Rossi, M. Sevaux, *Exact and metaheuristic approaches for memory cache management*, EURO XXIV, 24th European Conference on Operational Research, Lisbonne, Portugal, 11–14 juillet 2010.

CI19 A. Rossi, *On Designing a Robust Electricity Distribution Network*, EURO XXIV, 24th European Conference on Operational Research, Lisbonne, Portugal, 11–14 juillet 2010.

CI20 J-C. Créput, R. Dafali, A. Rossi, M. Sevaux, B. Zerbo *From simple heuristics to evolutionary approach for routing messages in a NoC*, EU/MEeting 2010, 10th anniversary of the metaheuristic community, Lorient, France 3–4 juin 2010.

CI21 M. Soto, A. Rossi, M. Sevaux, *Exact and metaheuristic approaches for a memory allocation problem*, EU/MEeting 2010, 10th anniversary of the metaheuristic community, Lorient, France, 3–4 juin 2010.

CI22 M. Sevaux, A. Rossi, K. Sörensen, *Electronic design : a new field of investigation for large scale optimization*, EU/MEeting 2009 Debating the future : new areas of application and innovative approaches, Porto, Portugal, 29–30 avril 2009. hal-00379957.

1.3.7 Conférences nationales (depuis 2006)

CF01 M. Soto, A. Rossi, M. Sevaux, *Allocation de mémoire dynamique dans les systèmes embarqués*, ROADEF 2012, Angers, France.

CF02 A. Rossi, *Une matheuristique pour l'arbre dominant de poids minimum*, ROADEF 2012, Angers, France.

CF03 F. Castaño, N. Velasco, A. Rossi, M. Sevaux, *Diversification and stabilization strategies to accelerate the convergence of the minimum coverage breach problem under bandwidth constraints in wireless sensor networks*, ROADEF 2012, Angers, France.

CF04 M. Soto, A. Rossi, M. Sevaux, *Synthèse sur l'allocation de mémoire dans les systèmes embarqués*, ROADEF 2011, pp. 195–196, Saint-Etienne, France.

CF05 M. Soto, A. Rossi, M. Sevaux, *Allocation de mémoire dynamique dans les systèmes embarqués*, MajecSTIC, octobre 2010.

CF06 M. Soto, A. Rossi, M. Sevaux, *Métaheuristiques pour l'allocation de mémoire dans les systèmes embarqués*, ROADEF 2010, pp. 35–43, Toulouse, France. (Taux d'acceptation : 48%).

CF07 G. Morel, C. Chauvin, A. Rossi, P. Berruet, *Concevoir des systèmes sociotechniques complexes résilients et reconfigurables pour garantir un niveau de sécurité optimal*, Ergo'IA 2008, pp. 145–148, Biarritz, France.

CF08 K. Trabelsi, P. Coussy, A. Rossi, M. Sevaux, *Ordonnancement et Assignation en Synthèse de Haut Niveau*, ROADEF 2008, pp. 341–342, Clermont-Ferrand, France.

CF09 A. Aubry, A. Rossi, M. Jacomino, *On finding a minimum-cost robust configuration for multipurpose machines*, ROADEF 2008, pp. 29–30, Clermont-Ferrand, France.

CF10 A. Belabbas, A. Rossi, P. Berruet, J-L. Philippe, *Ordonnancement prévisionnel en ligne pour une flotte de fauteuils roulants électriques*. Workshop International : Logistique et Transport 2007, Sousse, Novembre 2007.

CF11 A. Rossi, *Preuve de complexité pour une variante du problème de transport*. Conférence conjointe en recherche opérationnelle et aide à la décision FRANCORO V / ROADEF'07, pages 381–382, Grenoble, France.

CF12 M. Sevaux, A. Rossi. *Programmation linéaire en nombres entiers pour la synthèse de haut niveau*. Conférence conjointe en recherche opérationnelle et aide à la décision FRANCORO V / ROADEF'07, pages 399–400, Grenoble, France.

1.4 Activités d'enseignement

Ma philosophie de l'enseignement repose sur trois fondements. Le premier d'entre eux est bien sûr l'exemple que m'ont laissé mes maîtres au cours de ma formation, et qui continue de m'inspirer. Je dois le second fondement de ma vision de la pédagogie aux séminaires dispensés par le CIES de Grenoble, où de grands principes m'ont permis de donner du sens aux « ficelles » du métier que ces séminaires m'ont aussi permis d'acquérir. Enfin, ma propre expérience m'a conduit à remettre en question le principe de « transmission des connaissances ». Le troisième fondement de ma pratique pédagogique me conduit ainsi à penser que ce n'est pas le maître qui apprend la lecture aux élèves, mais bien les élèves qui apprennent la lecture avec le maître. Ainsi je ne construis pas mes interventions avec pour objectif de « transmettre », mais plutôt pour susciter la réflexion, et donc l'appropriation de connaissances par les étudiants. Je transmets du « matériau » exploitable et de qualité, mais je sais (et je le dis aux étudiants) qu'il leur appartient d'en faire des connaissances, et que cela passe par un travail d'appropriation que nul ne peut faire à leur place. Désireux d'encourager la réussite de tous, je reste disponible et ouvert à toutes les questions. Je fournis systématiquement aux étudiants une copie de l'examen de l'année précédente ainsi que le corrigé détaillé pour que chacun puisse se préparer dans les meilleures conditions. Dans le même esprit, je mets en ligne les corrections des devoirs à la maison et des examens dans des délais très courts (quelques jours).

Au plan comptable, ma charge d'enseignement se situe généralement autour de 250 heures équivalent TD par an. Cette section décrit les matières dans lesquelles j'interviens, le public, et les objectifs que je me fixe dans chaque cas.

1.4.1 Automatique continue

Public : Elèves ingénieurs de première année (ENSIBS).
Volume horaire : 8 h de cours, 8 h de TD, 8 h de TP.

L'objectif du cours est de permettre aux étudiants de s'approprier la culture scientifique et technique permettant de poser correctement le problème de la commande des systèmes continus. Le cours présente et illustre notamment les systèmes du premier et du second ordre, la stabilité, la synthèse de correcteurs (analytique et par la méthode de Ziegler-Nichols), le rejet de perturbation. J'insiste sur les hypothèses fondamentales (linéarité, invariance et causalité) à l'origine des résultats présentés, ce que j'appelle « la stratégie de calcul » et le recours aux méthodes d'ingénierie (décomposition des systèmes complexes, qualité rédactionnelle et vérification des résultats). Les TP sont effectués sous SCILAB, j'ai mis en ligne une démonstration de l'utilisation de XCOS disponible sous `http://www.youtube.com/watch?v=nKSvAX9D1Vc`. Un livret est remis aux étudiants en début de module avec une bibliographie, le sujet des TD, des devoirs à la maison, et des TP. Les étudiants y trouvent aussi l'examen de l'année précédente et sa solution détaillée.

1.4.2 Commande par retour d'état

Public : Etudiants en 1ère année de master Electronique et Informatique Industrielle (UBS).
Volume horaire : 8 h de cours, 6 h de TD, 12 h de TP.

Ce cours s'adresse à des étudiants connaissant l'automatique « classique », c'est-à-dire basée sur les fonctions de transfert. Il constitue une introduction à la commande par retour d'état, visant à mettre les étudiants en capacité de commander les systèmes linéaires par placement de pôle, avec ou sans observateur. Je m'attache à faciliter la compréhension des étudiants en attirant leur attention sur les outils mathématiques qu'ils doivent maîtriser avant le début du cours (théorème de Cayley-Hamilton, calcul de séries, méthode des cofacteurs pour l'inversion de matrices symboliques). Pendant le cours lui-même (dont le contenu est très technique), je multiplie les exemples en m'appuyant sur ce qu'ils connaissent (l'automatique « classique ») et en mettant en évidence ce que la représentation d'état permet d'apporter (avantages du retour d'état par rapport au retour de sortie, observabilité et commandabilité). Le placement de pôle et la synthèse d'observateur sont abordés dans le même esprit. Un livret est remis aux étudiants en début de module avec une bibliographie, les planches de cours incomplètes (à compléter en cours), le sujet des TD et des TP. Les étudiants y trouvent aussi l'examen de l'année précédente et sa solution détaillée. Tous les supports sont en anglais, mais le cours est donné en français.

1.4.3 Conception de commande industrielle répartie

Public : Elèves ingénieurs de deuxième année (ENSIBS, spécialité génie industriel).
Volume horaire : 2 h de cours, 8 h de TP.

Ce cours est divisé en trois parties, j'assure la première puis mes collègues Florent de Lamotte et Adel Baganne couvrent les aspects relatifs aux réseaux et à la supervision respectivement. La séance de cours me permet d'introduire le mécanisme d'appel réponse constituant le fondement de la commande hiérarchique entre grafcets. J'illustre mon propos par divers exemples de dysfonctionnements résultant d'une mauvaise synchronisation, je présente les erreurs classiques et j'introduis une méthode de décomposition permettant aux étudiants de préparer la commande d'un système de convoyage, qui fait l'objet des séances de TP sous PL7-PRO. Chaque groupe d'étudiants teste alors sa commande sur le convoyeur, et je les invite à tirer le maximum d'enseignements de chaque essai afin de minimiser la durée d'immobilisation du système de production (la conception de la commande étant effectuée hors-ligne). Plusieurs cahiers des charges de complexité croissante sont ainsi testés, afin d'aboutir à une commande hiérarchique basée sur un découpage en processus. L'étape suivante (distribution de la commande sur différents automates) est prise en charge par mes collègues. Un livret est remis aux étudiants en début de module avec une bibliographie et le sujet du TP. Une séance de démonstration du comportement attendu est organisée sur le système de convoyage avant les séances de TP afin de permettre aux étudiants de se faire une idée plus précise du système à commander.

1.4.4 Scheduling

Public : Elèves ingénieurs de deuxième année (ENSIBS, spécialité génie industriel).
Volume horaire : 4 h de cours, 6 h de TD, 4 h de TP.

Ce cours constitue une introduction à l'ordonnancement, qui fait l'objet d'un traitement plus approfondi en troisième année. Le cours de deuxième année introduit les notions de ressource, de tâche, de contrainte et d'objectif, dans des contextes qui dépassent le cadre de la production manufacturière. Après l'étude de problèmes faciles (au sens de la théorie de la complexité) à une machine, les problèmes à machines identiques sont présentés (heuristique LPT, algorithme de Mc Naughton pour le cas préemptif). Un exemple illustratif accompagne chaque problème, et les TD sont l'occasion de mettre en évidence certains aspects contre-intuitifs (étude d'ordonnancements au plus tôt, sans retard, anomalies de Graham sur le problème à machines identiques avec contraintes de précédence). La séance de TP permet aux étudiants de coder certains algorithmes en Visual Basic en appelant une fonction de tri qui leur est fournie. Un livret est remis aux étudiants en début de module avec une bibliographie, les planches de cours incomplètes (à compléter en cours), le sujet des TD et des TP. Les étudiants y trouvent aussi l'examen de l'année précédente et sa solution détaillée. Tous les supports sont en anglais, les cours, TD et TP sont dispensés en anglais.

1.4.5 Operations research

Public : Elèves ingénieurs de deuxième année (ENSIBS, spécialité génie industriel).
Volume horaire : 12 h de cours, 14 h de TD.

Ce cours constitue une introduction à la recherche opérationnelle, qui fait l'objet d'un traitement plus approfondi au second semestre, et en troisième année. Le cours de deuxième année est constitué d'une introduction à la théorie des graphes, couvrant les notions de connexité (faible et forte), la représentation des graphes (matrices d'adjacence, d'incidence), le calcul de plus courts chemins et de plus long chemin, et enfin les problèmes de flot maximum et coupe minimum. Chaque algorithme fait l'objet d'une présentation détaillée, puis d'un exemple d'application. La deuxième partie du cours est consacrée à la programmation linéaire. J'insiste sur la modélisation (qui occupe également une part importante en TD) et les « bonnes pratiques » (définition des variables de décision, unités physiques, rédaction des contraintes en langage naturel puis formel). La présentation de la méthode du Simplexe (phases I et II) est agrémentée d'exemples illustrant les points particuliers (coût réduit, choix des variables entrantes, sortantes, dégénérescence, cyclage, règle de Bland). Je veille à ce que les étudiants comprennent toutes les étapes, plutôt que d'appliquer une mécanique aveugle. La relation avec l'approche graphique est abondamment utilisée pour illustrer les points mentionnés ci-dessus. Le cours se termine par la construction d'un modèle pour le problème d'affectation, le problème de transport et le problème de transbordement. Un livret est remis aux étudiants en début de module avec une bibliographie, les planches de cours incomplètes (à compléter en cours), le sujet des TD et des

devoirs à la maison. Les étudiants y trouvent aussi l'examen de l'année précédente et sa solution détaillée. Tous les supports sont en anglais, les cours et les TD sont dispensés en anglais.

1.4.6 Petri nets

Public : Elèves ingénieurs de troisième année (ENSIBS, spécialité génie industriel).
Volume horaire : 6 h de cours, 10 h de TD, 8 h de TP.

L'objectif de ce cours est de développer des compétences avancées en modélisation des systèmes à événements discrets, mais également de produire des analyses pertinentes et rigoureuses des systèmes existants ou à concevoir. Après avoir présenté les points communs et les différences avec le Grafcet, je montre comment les réseaux de Petri apportent une réponse au problème de l'explosion combinatoire qui affecte les graphes d'état. J'introduis le formalisme matriciel associé à l'aspect graphique des réseaux de Petri, puis les « bonnes propriétés » et autres propriétés structurelles (P-semi flots et T-semi flots) sont illustrées par de nombreux exemples. A cette fin, je mets en ligne un document d'auto-formation que j'ai produit, visant une remise à niveau en algèbre linéaire pour que le calcul de noyau d'une matrice ne pose pas de difficulté aux étudiants. Le cours aborde aussi les réseaux de Petri T-temporisés, et les réseaux de Petri colorés. Les TD se focalisent sur la modélisation de systèmes de production à partir de ces différentes classes de réseaux de Petri. Les TP sont l'occasion d'utiliser deux logiciels libres, VISUAL OBJECT NET++ et TINA (développé par le LAAS à Toulouse), dans le cadre d'activités visant à dimensionner des stocks, et à contrôler une cellule robotisée virtuelle. Un livret est remis aux étudiants en début de module avec une bibliographie, les planches de cours incomplètes (à compléter en cours), le sujet des TD, des TP et des devoirs à la maison. Les étudiants y trouvent aussi l'examen de l'année précédente et sa solution détaillée. Tous les supports sont en anglais, les cours, TD et TP sont dispensés en anglais.

1.4.7 Advanced scheduling

Public : Elèves ingénieurs de troisième année (ENSIBS, spécialité génie industriel).
Volume horaire : 8 h de cours, 10 h de TD, 8 h de TP.

Ce cours fait suite à l'introduction à l'ordonnancement dispensée en deuxième année. Il présente plusieurs méthodes arborescentes pour aborder des problèmes difficiles à une machine, et illustre l'intérêt des relaxations pour calculer des bornes inférieures augmentant l'efficacité des méthodes arborescentes. La seconde partie du cours présente les problèmes d'atelier (*open shop*, *flowshop*, puis *jobshop*), des résultats de complexité, des propriétés de régularité de critères et de dominance d'ensembles de solutions, mais aussi des heuristiques (algorithme de Mitten, heuristique de Campbell Dudek et Smith) et des algorithmes bien connus (LAPT, Johnson, Jackson). On s'efforce d'illustrer ces notions en TD, et les TP tirent profit des connaissances en modélisation des étudiants afin de comparer les méthodes de résolution présentées en cours et en TD avec les résultats d'un solver (XPRESS-IVE et/ou le solver incorporé à Microsoft Excel et Libre Office). Un livret est remis aux étudiants en début de module avec une bibliographie, les planches de cours incomplètes (à compléter en cours), le sujet des TD et des TP. Les étudiants y trouvent aussi l'examen de l'année précédente et sa solution détaillée. Tous les supports sont en anglais, les cours, TD et TP sont dispensés en anglais.

1.4.8 Recherche opérationnelle

Public : Etudiants en 2$^{\text{ème}}$ année de master Gestion et Pilotage de la Production (formation continue, UBS).
Volume horaire : 8 h de cours, 4 h de TD, 4 h de TP.

Ce cours d'ordonnancement couvre les mêmes sujets que le cours intitulé *Scheduling* dispensé en deuxième année d'école d'ingénieur, mais ses objectifs et son déroulement sont différents car le public est constitué d'étudiants en reprise d'études, pour lesquels les mathématiques constituent souvent une difficulté de taille. Afin de rendre le cours plus accessible, j'ai confectionné une série d'exercices permettant aux étudiants de s'auto-évaluer, et de trouver des références bibliographiques et en ligne leur permettant de combler leur retard avant le début du cours de recherche opérationnelle. Ce public se distingue par une maturité et une motivation supérieure à celle des élèves ingénieurs, ce qui permet une interaction plus forte en classe et en dehors (communication par courrier électronique). Comme le niveau d'anglais des étudiants est souvent insuffisant ou très hétérogène, et afin de ne pas cumuler les difficultés, je dispense tous mes cours de formation continue en français, même si les supports sont eux, en anglais.

1.4.9 Réseaux de Petri

Public : Etudiants en 2ème année de master Gestion et Pilotage de la Production (formation continue, UBS).
Volume horaire : 8 h de cours, 4 h de TP.

Ce cours de réseaux de Petri est le pendant du cours intitulé *Petri nets* dispensé en deuxième année d'école d'ingénieur, mais il est adapté au public de formation continue (ainsi qu'au volume horaire plus restreint). Les réseaux de Petri colorés ne sont pas abordés en classe, et seul le logiciel VISUAL OBJECT NET++ est utilisé en TP. L'objectif est moins ambitieux sur le plan académique, et l'accent est mis sur la modélisation, ce qu'on peut tirer d'un modèle formel, et la comparaison avec les résultats de la simulation par exemple. Comme le cours de recherche opérationnelle, il participe à la remise à niveau en sciences de professionnels ayant quitté l'université depuis parfois très longtemps, et qui voient ainsi leur aptitude au raisonnement et à l'abstraction augmenter au cours de la formation.

1.4.10 Ordonnancement

Public : Etudiants en 2ème année de master Gestion et Pilotage de la Production (formation continue, UBS).
Volume horaire : 8 h de cours, 4 h de TD, 4 h de TP.

Ce cours correspond aux modules *Scheduling* et *Advanced scheduling* dispensés en deuxième année d'école d'ingénieur, mais il est adapté au public de formation continue : seuls les problèmes faciles (au sens de la théorie de la complexité) sont abordés, et l'objectif est d'offrir un panorama des problèmes qui peuvent se poser dans l'industrie, et des moyens de les résoudre. La résolution des problèmes en séance de TP exploite les compétences en modélisation acquises dans le module de recherche opérationnelle, et permet aux étudiants d'aborder des problèmes difficiles à l'aide d'un solver.

1.4.11 Autres activités pédagogiques

Je complète mon service avec diverses activités de tutorat, et des encadrements de stages de fin d'études. Il m'arrive occasionnellement de donner des cours sur les automates programmables industriels en licence énergétique à l'UBS, des cours d'algorithmique à l'IUT de Lorient (département QLIO), et des cours d'optimisation combinatoire en troisième année d'école d'ingénieur à l'ENSIBS. Mon attachement à la formation d'ingénieur se traduit aussi par ma participation depuis 2011 aux jurys officiels de l'épreuve de travaux d'initiative personnelle encadrés (TIPE), qui font partie des Concours Communs Polytechniques. Je suis interrogateur en sciences physiques et en sciences pour l'ingénieur, dans les sections PSI et TSI.

J'ai dispensé à Grenoble des cours d'ordonnancement de 2004 à 2006 à des étudiants de l'université de Monterrey (Mexique). J'ai également créé un cours de 6 heures sur la résolution de programmes linéaires en variables mixtes (méthodes arborescentes, coupes de Gomory, inégalités MIR et branch-and-cut) que je dispense en Master of Science (M.Sc) à l'université d'Hyderabad depuis 2009.

Chapitre 2

Optimisation de la qualité de service et de la durée de vie des réseaux de capteurs sans fil

Résumé

Ce chapitre aborde trois problèmes relatifs à l'exploitation des réseaux de capteurs sans fil soumis à une contrainte de bande passante. Le premier consiste à minimiser le défaut de couverture global tout en garantissant que la durée de vie du réseau reste supérieure à une borne minimale donnée. Un défaut de couverture apparaît dans le réseau lorsque certaines cibles ne sont couvertes par aucun capteur actif. Le second problème consiste à maximiser la durée de vie du réseau tout en garantissant que le défaut de couverture global reste inférieur à une borne maximale donnée. Enfin le troisième problème explore les compromis entre défaut de couverture global et durée de vie du réseau. Une méthode exacte basée sur la génération de colonnes est proposée pour les trois problèmes, ainsi qu'une heuristique simple dérivée de l'algorithme exact. L'utilisation d'un algorithme génétique pour produire des colonnes profitables permet de diminuer très sensiblement les temps de calcul.

2.1 Introduction

L'utilisation des réseaux de capteurs sans fil s'est généralisée pour la surveillance et la collecte d'informations en milieu hostile [2]. Les domaines d'application les plus courants de ces réseaux particuliers sont la surveillance des champs de bataille, la détection anticipée des incendies de forêt et les dispositifs d'alerte sous-marins des tsunamis, entre autres [57, 73]. Bien souvent, les capteurs constituant le réseau ne peuvent être positionnés avec précision, et se trouvent disséminés de manière erratique dans la zone à surveiller. Afin de compenser le caractère aléatoire de leur placement, un grand nombre de capteurs est généralement déployé, ce qui contribue également à augmenter la tolérance aux fautes du réseau car les cibles peuvent se trouver à portée de plusieurs capteurs. La baisse des coûts consécutive aux progrès de l'électronique, des transmissions radio et des systèmes embraqués permet ainsi le déploiement de réseaux de capteurs sans fil constitués d'un nombre croissant de nœuds. Chaque capteur est alimenté par une batterie dont l'autonomie est limitée. La durée de vie du réseau de capteurs dépend alors étroitement de la manière dont l'énergie disponible est utilisée, le renouvellement des batteries étant exclu en raison du caractère hostile ou peu accessible de l'environnement. La technique d'optimisation de la durée de vie du réseau la plus répandue tire parti de la redondance offerte par le grand nombre de capteurs. Ceux-ci sont regroupés dans des sous-ensembles non nécessairement disjoints appelés *couvertures*, de façon à ce que tous les capteurs faisant partie d'une même couverture couvrent toutes les cibles. Les couvertures sont alors activées de façon à ce qu'une et une seule couverture soit active à tout instant, jusqu'à la fin de la durée de vie du réseau. Activer une couverture signifie que tous les capteurs qu'elle contient sont actifs, alors que les autres capteurs sont en veille. L'utilisation de couvertures est efficace pour prolonger la durée de vie du réseau pour deux raisons principales. La première est que les capteurs consomment considérablement plus d'énergie lorsqu'ils sont actifs que lorsqu'ils sont en veille [54]. La seconde raison tient à la technologie des accumulateurs. Les études dans ce domaine ont montré que l'autonomie d'une batterie peut doubler si son utilisation est fractionnée plutôt que continue [9, 10].

Dans l'exemple suivant, tiré de [69], trois capteurs notés s_1, s_2 et s_3 sont chargés de couvrir (c'est-à-dire de surveiller) trois cibles notées τ_1, τ_2 et τ_3. Chaque capteur ne peut être actif que pendant une unité de temps, cette durée pouvant être fractionnée à volonté. La zone de couverture d'une capteur est un disque centré sur ce capteur, dont le périmètre est représenté en traits pointillés sur la figure 2.1. On peut voir que τ_1 peut être couverte par les capteurs s_1 et s_3, τ_2 peut être couverte par s_1 et s_2, et τ_3 peut être couverte par s_2 et s_3. A tout instant, chacune des trois cibles doit être couverte par au moins un capteur actif : le défaut de couverture est interdit dans cet exemple.

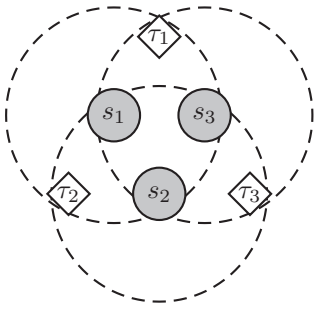
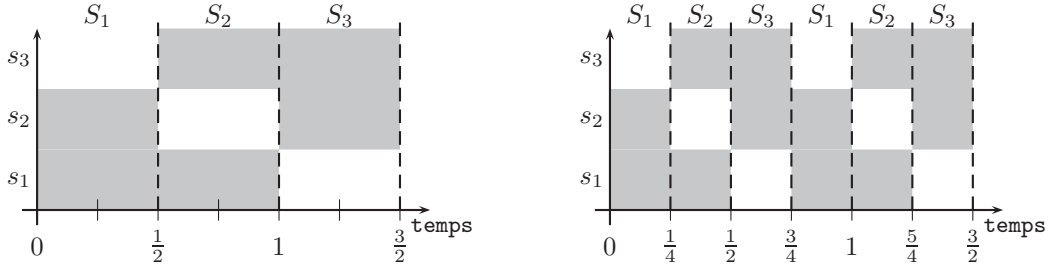


FIGURE 2.1 – Exemple de réseau sans fil contenant trois capteurs, et couvrant trois cibles.

Si l'on impose que les couvertures soient disjointes dans l'exemple de la figure 2.1, alors on peut facilement déduire que la durée de vie maximale du réseau est d'une unité de temps. En effet, deux capteurs doivent être utilisés simultanément pour couvrir toutes les cibles, et une fois épuisées les batteries de ces capteurs (soit après une unité de temps), le capteur restant ne permet pas, seul, de couvrir toutes les cibles.

Si les couvertures ne sont plus tenues d'être disjointes, la solution optimale consiste à utiliser successivement les trois couvertures S_1, S_2 et S_3 suivantes, pour une durée de 0.5 unité de temps chacune. $S_1 = \{s_1, s_2\}$, $S_2 = \{s_1, s_3\}$ et $S_3 = \{s_2, s_3\}$, ce qui porte la durée de vie du réseau à 1.5 unités de temps.

On peut envisager la mise en œuvre de cette dernière solution de plusieurs manières, deux ordonnancements sont donnés sur la figure 2.2 à titre d'exemple. L'ordonnancement le plus simple consiste à utiliser la couverture S_j continûment pour t_j unités de temps (figure 2.2(a)), où t_j est ici égal à 0.5 unités de temps. L'interruption de l'utilisation d'une couverture est néanmoins possible, sinon souhaitable d'après [9, 10]. On peut alors créer η copies de chaque couverture, chacune d'elles étant utilisée $\frac{t_j}{\eta}$ unités de temps. La figure 2.2(b), illustre cette solution pour $\eta = 2$. Le choix de la valeur de η relève de la technologie des accumulateurs, et n'est pas abordé ici. La question de l'ordonnancement des couvertures sans préemption a néanmoins été abordée dans [59].



(a) Un ordonnancement sans préemption

(b) Un ordonnancement pour $\eta = 2$

FIGURE 2.2 – Deux manières d'ordonnancer les couvertures.

En plus de la contrainte concernant la constitution des couvertures, on suppose que des contraintes relatives à la bande passante utilisée par les capteurs pour transmettre le résultat de leurs mesures en temps réel doivent être satisfaites. On appelle *observateur* l'appareil chargé de collecter toutes les données transmises par les capteurs, de les traiter, et de communiquer avec le monde extérieur. L'observateur est supposé être à portée de communication de tous les capteurs. Cependant, en raison des limites sur la bande passante, certains capteurs actifs peuvent se trouver dans l'impossibilité de communiquer avec l'observateur, faute de canal disponible. Si W est le nombre maximal de canaux de communication disponibles (et utilisables simultanément), alors cela implique que le nombre de capteurs dans toute couverture ne peut dépasser W. Ce mode de prise en compte des limites de bande passante est introduit dans [15]. Limiter la taille des couvertures peut cependant conduire à l'impossibilité de couvrir toutes les cibles, ou à ne pouvoir assurer une couverture totale que pendant une durée très faible. Pour un réseau de capteurs sans fil et un ensemble de cibles donnés, le problème de minimisation du défaut de couverture global sous contrainte de bande passante consiste à former des couvertures constituées d'au plus W capteurs, de manière à ce que la durée de vie du réseau demeure supérieure à une durée donnée, et que la somme des durées de non-couverture de toutes les cibles, soit minimum. Les auteurs de [15] ont montré que ce problème était $\mathcal{NP}$-difficile, et ont proposé une formulation à base de programmation linéaire en nombres entiers et deux heuristiques pour le résoudre. Cependant, ils ont limité le nombre de couvertures à $\lceil \frac{n}{W} \rceil$, où n désigne le nombre total de capteurs dans le réseau, ce qui induit une borne supérieure assez contraignante sur la durée de vie du réseau.

Plus récemment, la contrainte imposant que les couvertures soient disjointes a été remise en question dans [69], les auteurs ayant montré qu'elle n'était pas justifiée dans ce contexte. Ils ont proposé de résoudre le problème du défaut de couverture global et de la durée de vie du réseau de capteurs sans fil sous contrainte de bande passante en abordant conjointement l'optimisation de l'énergie disponible et l'utilisation de la bande passante, et ont introduit les deux problèmes suivants. Le premier consiste à minimiser le défaut de couverture sous contrainte de bande passante tout en assurant que la durée de vie du réseau reste supérieure à une durée donnée, il est appelé MCBB pour *minimum coverage breach under bandwidth constraints*. Le second problème vise à maximiser la durée de vie du réseau de capteurs tout en garantissant que le défaut de couverture global reste inférieur à un seuil donné afin d'offrir une qualité de service acceptable, ce problème est appelé MNLB, pour *maximum network lifetime under bandwidth constraints*. MCBB est adapté aux situations pour lesquelles la durée de vie du réseau est critique, alors que MNLB est plus pertinent lorsque la qualité de service est critique.

Le première heuristique proposée dans [69] résout la relaxation continue de la formulation du problème par

programmation linéaire en nombres entiers, et répare la solution fractionnaire ainsi obtenue pour la rendre faisable pour MCBB. La seconde heuristique est un algorithme glouton qui construit des couvertures devant être utilisées pendant la même durée. D'après [69], MNLB est plus difficile que MCBB car le défaut de couverture est global, autorisant la compensation du défaut de couverture d'une couverture particulière par une autre. Ainsi, l'algorithme glouton proposé dans [69], qui construit les couvertures sur la seule base de leur défaut de couverture individuel, ne peut retourner de bonnes solutions faute de pouvoir exploiter la compensation entre couvertures.

Plus généralement, les techniques de construction de couvertures pour les réseaux de capteurs sans fil [13, 69] reposent sur l'approche en deux temps suivante. D'abord, un modèle mathématique non linéaire du problème est linéarisé par l'introduction de nouvelles variables, et sa relaxation continue est résolue. Ensuite, une fonction de réparation est utilisée pour produire une solution faisable. Or cette approche repose sur un modèle qui peut ne pas refléter fidèlement la réalité en raison de la linéarisation et de la relaxation des contraintes d'intégrité opérées sur la formulation du problème original, sans compter que la fonction de réparation introduit un biais supplémentaire.

La génération de colonnes [45] fournit un moyen efficace d'obtenir des solutions exactes sans avoir à travailler avec une formulation non linéaire du problème. Par ailleurs, la génération de colonnes a déjà été employée avec succès pour des problèmes de réseaux de capteurs [3]. Cette approche peut cependant être lente pour plusieurs raisons, comme la dégénérescence de la solution optimale [4], l'effet plateau [67] (démontrer l'optimalité d'une solution peut nécessiter un très grand nombre d'itérations), et le fait qu'un problème difficile doive être résolu à chaque itération. Ces inconvénients peuvent être contournés dans une certaine mesure en utilisant un algorithme génétique pour produire plusieurs colonnes profitables à chaque itération, puis une méthode exacte appelée lorsque l'algorithme génétique ne parvient plus à retourner de colonne profitable. Le résultat est un algorithme hybride basé sur la génération de colonnes exploitant une métaheuristique pour accélérer la convergence. Les approches proposées dans ce chapitre relèvent de la catégorie des *matheuristiques* basées sur la génération de colonnes [27, 52, 53]. La première contribution ayant fait date dans ce domaine est dûe à Filho and Lorena [27]. Ils ont utilisé un algorithme génétique dans un algorithme de génération de colonnes pour résoudre un problème de coloration de graphe. Mais comme le problème auxiliaire (ou sous-problème) n'est abordé qu'avec un algorithme génétique, aucune garantie d'optimalité ne peut être fournie par cette approche. Puchinger et Raidl [52, 53] ont été les premiers à proposer une approche à plusieurs niveaux pour résoudre le problème auxiliaire, afin de garantir l'optimalité de la solution obtenue par l'algorithme de génération de colonnes. Plus précisément, ils ont proposé une approche pour le problème du *bin packing* bidimensionnel à trois niveaux. D'abord, une heuristique gloutonne est appelée pour tenter de résoudre le problème auxiliaire. En cas d'échec, un algorithme génétique prend le relai pour aborder une version simplifiée du problème auxiliaire. En cas de nouvel échec, une formulation par programmation linéaire en nombres entiers d'une version simplifiée du problème auxiliaire est résolue à l'aide d'un solver. Enfin, si les trois tentatives précédentes ont été infructueuses, la formulation du problème auxiliaire non simplifiée fait l'objet d'une résolution par le solver.

Ce chapitre est organisé comme suit. La section 2.2 définit plus formellement MCBB et présente l'algorithme de résolution basé sur la génération de colonnes. La section 2.3 est dévolue à MNLB pour lequel une heuristique est également proposée. Le problème de l'exploration des compromis entre défaut de couverture et durée de vie, noté TOLB fait l'objet de la section 2.4, et la section 2.5 clôt ce chapitre.

2.2 Minimisation du défaut de couverture global (MCBB)

2.2.1 Définition du problème

Le problème de minimisation du défaut de couverture global sous contrainte de bande passante (MCBB) consiste à construire et à ordonnancer des couvertures de façon à ce que la durée de vie du réseau de capteurs reste supérieure à une durée T_0 donnée. Le réseau est constitué de n capteurs identiques équipés de batteries elles aussi identiques. Sans perte de généralité, l'autonomie des batterie est normalisée à une unité de temps. Un capteur s couvre une cible si la distance qui les sépare est inférieure ou égale au rayon R_s définissant la zone de couverture du capteur. On note m le nombre de cibles, et C_k est l'ensemble des capteurs couvrant la cible k pour tout k dans $\{1, \ldots, m\}$.

Une couverture valide S_j est un sous-ensemble de capteurs dont la cardinalité est inférieure ou égale à W pour tout j appartenant à $\{1, \ldots, p\}$ (où p est le nombre de couvertures). Ces couvertures doivent aussi être ordonnancés, et on se limitera ici à calculer la durée t_j pendant laquelle S_j est active. On rappelle que deux couvertures ne peuvent être actives au même instant, et qu'exactement une couverture doit être active pendant l'intervalle de temps $[0, T_0]$, où T_0 est la durée de vie minimale du réseau. Ce problème admet une solution si et seulement is $T_0 \leq n$: la durée de vie maximale est atteinte lorsque chaque capteur est utilisé seul (le défaut de couverture global est également maximal dans ce cas).

Le défaut de couverture d'une cible est le temps total pendant lequel elle n'est couverte par aucun capteur actif

au cours de vie du réseau. Il est à distinguer du défaut de couverture d'une couverture, noté b_j, qui représente le nombre de cibles qu'aucun capteur de S_j ne couvre.

Dans [69], MCBB est formulé par un modèle mathématique non-linéaire, où :

- $x_{i,j}$ est une variable de décision binaire fixée à un si et seulement si le capteur i appartient à la couverture S_j
- $z_{j,k}$ est une variable de décision binaire fixée à un si et seulement si au moins un capteur de S_j couvre la cible k
- t_j est une variables continue non négative qui représente la durée pendant laquelle la couverture S_j est active.

$$\text{Minimiser} \quad \sum_{j=1}^{p}\sum_{k=1}^{m}(t_j - t_j z_{j,k}) \tag{2.1}$$

$$\text{Sous les contraintes} \quad \sum_{j=1}^{p} x_{i,j}t_j \leq 1 \qquad \forall i \in \{1,\ldots,n\} \tag{2.2}$$

$$\sum_{i\in C_k} x_{i,j} \geq z_{j,k} \qquad \forall k \in \{1,\ldots,m\}\forall j \in \{1,\ldots,p\} \tag{2.3}$$

$$\sum_{i=1}^{n} x_{i,j} \leq W \qquad \forall j \in \{1,\ldots,p\} \tag{2.4}$$

$$\sum_{j=1}^{p} t_j \geq T_0 \tag{2.5}$$

$$x_{i,j} \in \{0,1\} \qquad \forall i \in \{1,\ldots,n\}\forall j \in \{1,\ldots,p\} \tag{2.6}$$

$$z_{j,k} \in \{0,1\} \qquad \forall j \in \{1,\ldots,p\}\forall k \in \{1,\ldots,m\} \tag{2.7}$$

$$t_j \geq 0 \qquad \forall j \in \{1,\ldots,p\} \tag{2.8}$$

La fonction objectif vise à minimiser le défaut de couverture global. La contrainte. La contrainte (2.2) assure qu'aucun capteur n'est utilisé plus d'une unité de temps (en raison de l'autonomie limitée de sa batterie), (2.3) stipule qu'une cible est couverte par une couverture si l'un des capteurs appartenant à cette couverture couvre cette cible, et les contraintes $(2.4) - (2.5)$ font respecter la limite de bande passante disponible et la durée de vie minimale du réseau à assurer.

2.2.2 Formulation de MCBB par programmation linéaire en variables mixtes

Le modèle mathématique non linéaire de la section précédente est linéarisé à l'aide des variables et des contraintes supplémentaires présentées ci-après.

- Pour tout (j,k) dans $\{1,\ldots,p\}\times\{1,\ldots,m\}$, la variable $q_{j,k}$ est introduite pour remplacer $t_j z_{j,k}$ dans l'équation (2.1), ce qui conduit à la fonction objectif linéarisée suivante :

$$\text{Minimiser} \quad \sum_{j=1}^{p}\sum_{k=1}^{m}(t_j - q_{j,k}) \tag{2.9}$$

- Pour tout (i,j) dans $\{1,\ldots,n\} \times \{1,\ldots,p\}$, la variable $r_{i,j}$ est introduite pour remplacer $t_j x_{i,j}$ dans la contrainte (2.2), ce qui conduit à la contrainte linéaire suivante :

$$\sum_{j=1}^{p} r_{i,j} \leq 1 \quad \forall i \in \{1,\ldots,n\} \tag{2.10}$$

Ces nouvelles variables sont reliées aux variables originales par les contraintes suivantes :

$$t_j \leq t_{j+1} \qquad \forall j \in \{1, \ldots, p-1\} \tag{2.11}$$
$$t_p \leq 1 \tag{2.12}$$
$$\sum_{j=1}^{p} q_{j,k} \leq |C_k| \qquad \forall k \in \{1, \ldots, m\} \tag{2.13}$$
$$q_{j,k} \leq t_j \qquad \forall j \in \{1, \ldots, p\} \forall k \in \{1, \ldots, m\} \tag{2.14}$$
$$q_{j,k} \leq z_{j,k} \qquad \forall j \in \{1, \ldots, p\} \forall k \in \{1, \ldots, m\} \tag{2.15}$$
$$r_{i,j} \geq x_{i,j} + t_j - 1 \qquad \forall i \in \{1, \ldots, n\} \forall j \in \{1, \ldots, p\} \tag{2.16}$$
$$q_{j,k} \geq 0 \qquad \forall j \in \{1, \ldots, p\} \forall k \in \{1, \ldots, m\} \tag{2.17}$$
$$r_{i,j} \geq 0 \qquad \forall i \in \{1, \ldots, n\} \forall j \in \{1, \ldots, p\} \tag{2.18}$$

Les contraintes (2.11) et (2.12) visent à limiter la symétrie de la formulation précédente qui résulte de la numérotation (arbitraire) des couvertures. La contrainte (2.13) est une inégalité valide dont la présence a pour effet de réduire l'espace des solutions et de faire diminuer sensiblement le temps de calcul. Les contraintes $(2.14) - (2.18)$ sont induites par le rôle des variables $q_{j,k}$ et $r_{i,j}$ dans la linéarisation de la formulation.

Enfin, le programme linéaire en variables mixtes modélisant MCBB est appelé $MILP$, il est constitué des équations $(2.3)-(2.6)$ et $(2.8)-(2.18)$. La contrainte (2.7) peut être supprimée sans conséquence, car les variables $z_{j,k}$ seront nécessairement à valeurs binaires si les variables $x_{i,j}$ le sont aussi.

Cette formulation a été résolue avec XPRESS-MP [72], puis ses résultats comparés à ceux de l'algorithme basé sur la génération de colonnes présenté à la section 2.3, dans sa forme la plus simple (c'est-à-dire que le problème auxiliaire est résolu par programmation linéaire en variables entières, sans algorithme génétique). Pour des instances de taille très modeste ($n = 25$ capteurs, $m = 30$ cibles et $W = 5$), $MILP$ est résolu en 3.28 secondes, alors que l'algorithme basé sur la génération de colonnes résout la même instance en 0.34 seconde. Pour les plus petites instances utilisées dans la section 2.2.4, ($n = 50$ capteurs et $m = 30$ cibles), $MILP$ est résolu en 56.88 secondes, alors que l'algorithme basé sur la génération de colonnes résout la même instance en 0.88 seconde. Avec $n = 100$ capteurs et $m = 60$ cibles, aucune solution optimale n'est trouvée par XPRESS-MP pour $MILP$ après une heure de recherche alors que l'algorithme basé sur la génération de colonnes obtient la solution optimale en moins de 4 secondes. Cela montre que la génération de colonnes est particulièrement bien adaptée au problème, de sorte que la formulation $MILP$ ne sera plus utilisée.

2.2.3 Algorithme de génération de colonnes

MCBB est décomposé en deux sous-problèmes comme suit. Le problème maître consiste à calculer la durée d'utilisation des couvertures, et le problème auxiliaire est chargé de construire une (ou plusieurs) couvertures profitables supplémentaires pour le problème maître. La section 2.2.3 présente le problème maître en détail, ainsi que l'algorithme de génération de colonnes. La section 2.2.3 présente une approche exacte et un algorithme génétique pour aborder le problème auxiliaire.

Problème maître

MCBB peut être modélisé par le programme linéaire MAS_{MCBB} présenté ci-dessous. La couverture S_j est alors représentée par la colonne $[a_{1,j}, a_{2,j}, \ldots, a_{n,j}]^\top$ où $a_{i,j} = 1$ si le capteur i appartient à la couverture S_j, et $a_{i,j} = 0$ sinon. MAS_{MCBB} a p variables continues notées t_j.

$$\text{Minimiser} \quad \sum_{j=1}^{p} b_j t_j \tag{2.19}$$
$$\text{Sous les contraintes} \quad \sum_{j=1}^{p} a_{i,j} t_j \leq 1 \quad \forall i \in \{1, \ldots, n\} \qquad [y_i] \tag{2.20}$$
$$\sum_{j=1}^{p} t_j \geq T_0 \qquad [y_{n+1}] \tag{2.21}$$
$$t_j \geq 0 \qquad \forall j \in \{1, \ldots, p\} \tag{2.22}$$

La fonction objectif (2.19) vise à minimiser le défaut de couverture global, qui est la somme des durées de non-couverture de toutes les cibles. On rappelle que b_j désigne le défaut de couverture de la couverture S_j. L'ensemble de contraintes (2.20) assure que chaque capteur n'est pas utilisé plus d'une unité de temps, la variable duale associée à la contrainte portant sur le capteur i est notée y_i pour tout i dans $\{1, \ldots, n\}$. La contrainte (2.21) assure que la durée de vie du réseau est supérieure ou égale à T_0, la variable duale qui lui est associée est y_{n+1}.

Cette formulation est compacte, mais elle ne peut pas être exploitée en l'état car tout ensemble comptant au plus W capteurs constitue une couverture valide, ce qui implique que les couvertures sont au nombre de $p = \sum_{z=0}^{W} \binom{n}{z}$. A titre d'exemple, les plus petites instances considérées ici comptent $n = 50$ capteurs. $W = 5$ conduit à plus de deux millions de couvertures valides. Fort heureusement, il n'est pas nécessaire d'énumérer toutes les couvertures valides car toute base réalisable de MAS_{MCBB} compte au maximum $n + 1$ variables non nulles (ou moins s'il existe une base optimale dégénérée) puisque MAS_{MCBB} est constitué de $n + 1$ contraintes hors contraintes de non négativité des variables de décision. La génération de colonnes est une approche tout à fait appropriée à une telle situation : on commence par construire une version simplifiée de MAS_{MCBB}, en ne conservant qu'un nombre raisonnablement faible de colonnes. Cette version simplifiée est appelée *problème maître*. Ensuite, un processus itératif commence. Il consiste à résoudre le problème maître, puis à chercher à savoir si l'introduction d'une colonne dans la formulation courante du problème maître permettrait d'améliorer sa solution courante. Une telle colonne est dite *profitable*. Le problème auxiliaire est chargé de construire une ou plusieurs colonnes profitables. Ces colonnes sont alors ajoutées au problème maître et une nouvelle itération commence. Si aucune colonne profitable ne peut être trouvée (et qu'on peut apporter la preuve qu'il n'en existe aucune), alors la solution courante du problème maître est optimale et l'algorithme s'arrête. A partir de maintenant, p désigne le nombre de colonnes du problème maître, c'est-à-dire le nombre de couvertures valides explicitement présentes dans la formulation du problème maître à une itération donnée. Le problème auxiliaire est décrit en détail dans la section 2.2.3.

La génération de colonnes s'est avérée très efficace pour résoudre un grand nombre de problèmes [55], en particulier des problèmes de découpe [20, 32], d'ordonnancement et de constitution d'équipages [12], et d'optimisation pour les réseaux de capteurs sans fil [3]. A la différence de ces travaux, l'approche proposé ici associe un algorithme génétique à la génération de colonnes afin d'en accroître l'efficacité.

Les grandes étapes de l'approche proposée sont présentées sous la forme d'un logigramme à la figure 2.3. Initialement, seules les couvertures ne contenant qu'un seul capteur sont présentes dans le problème maître. Ce programme linéaire est alors résolu et le vecteur des variables duales $y^\top = [y_1, \ldots, y_{n+1}]$ est utilisé dans la fonction objectif du problème auxiliaire (voir section 2.2.3). En effet, le problème auxiliaire cherche à construire la colonne la plus profitable possible (c'est-à-dire celle qui est susceptible de contribuer à la plus forte amélioration de l'objectif du problème maître). Cette colonne est la couverture S_{p+1} qui minimise $b_{p+1} - y^\top A_{p+1}$ (voir [16, 71]). Cette colonne profitable est représentée par les $n + 1$ premiers éléments $a_{i,p+1}$ de la colonne A_{p+1} ci-dessous, qui sera ajoutée au problème maître au début de l'itération suivante.

$$A_{p+1} = \begin{bmatrix} a_{1,p+1} \\ a_{2,p+1} \\ \vdots \\ a_{n,p+1} \\ 1 \end{bmatrix}$$

Dans l'approche proposée, le problème auxiliaire est abordé pour la première fois avec un algorithme génétique appelé GA, qui peut être lancé jusqu'à quatre fois dans la même itération. GA est d'abord lancé pour N itérations, puis pour $N' > N$ itérations les fois suivantes. Si GA est incapable de trouver une couverture profitable, la formulation du problème auxiliaire basée sur la programmation linéaire en nombres entiers, appelée ILP, est résolue avec un solver. Le résultat est soit une couverture profitable, soit la preuve qu'il n'existe pas de telle couverture. On peut concevoir une heuristique en supprimant la résolution de ILP, et en retournant la solution du problème maître lorsque GA ne parvient plus à trouver de couverture profitable. Les performances de cette heuristique sont discutées à la section 2.3.6.

Problème auxiliaire

Le problème auxiliaire vise à construire une couverture profitable S_{p+1} pour le problème maître, c'est-à-dire qu'il calcule $a_{i,p+1} \in \{0, 1\}$ pour tout i dans $\{1, \ldots, n\}$ de manière à minimiser le coût réduit de la colonne correspondante à ajouter au problème maître tout en satisfaisant la contrainte de bande passante. Le problème auxiliaire est d'abord formulé à l'aide d'un programme linéaire en nombres entiers, puis un algorithme génétique que l'on désignera par GA est proposé pour générer une ou plusieurs couvertures profitables.

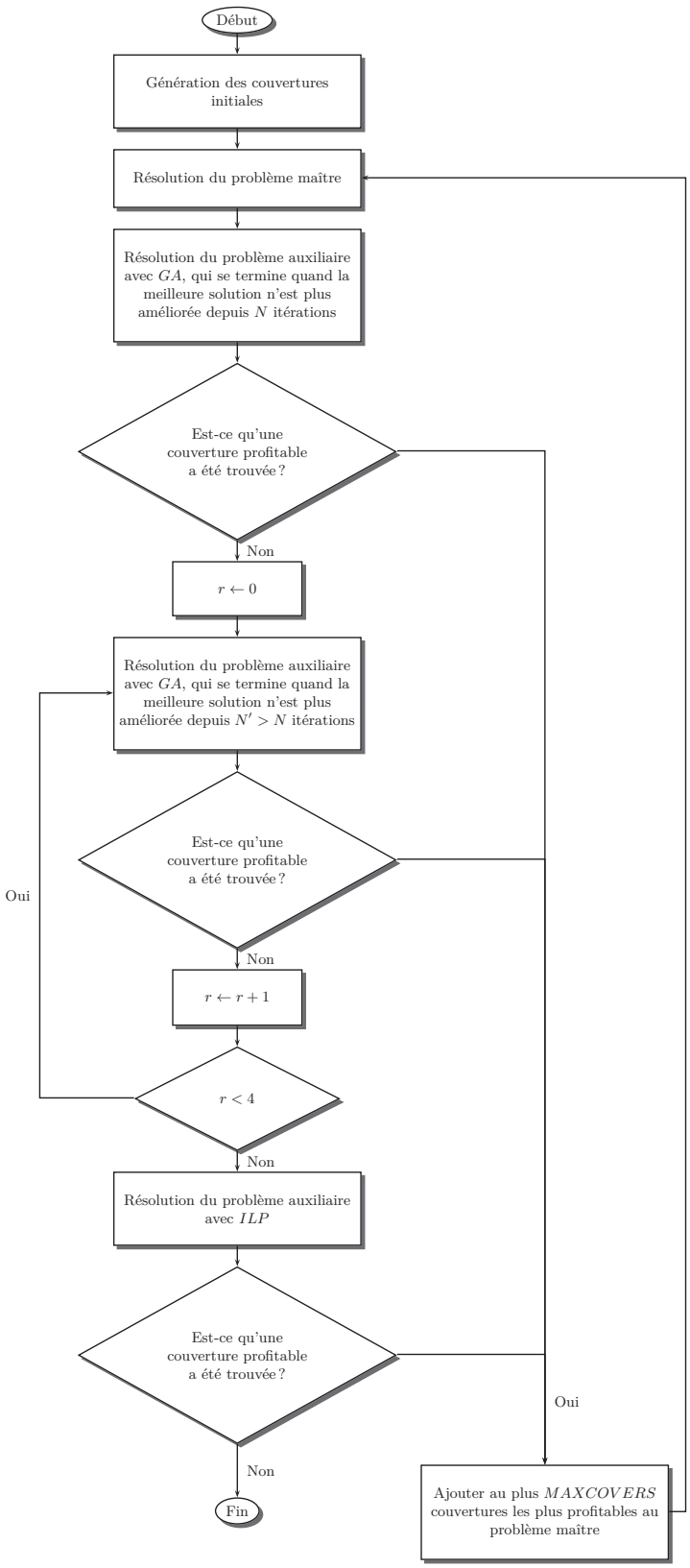


FIGURE 2.3 – Algorithme de génération de colonnes pour MCBB.

Formulation du problème auxiliaire sous la forme d'un programme linéaire en nombres entiers Cette formulation est notée AUX_{MCBB}. En plus de b_{p+1} et de $a_{i,p+1}$, r_k est une variable de décision définie par $r_k = 1$ si la cible k est couverte par un capteur de la couverture S_{p+1}, $r_k = 0$ sinon.

$$\text{Minimiser} \quad b_{p+1} - \sum_{i=1}^{n} y_i a_{i,p+1} - y_{n+1} \tag{2.23}$$

$$\text{Sous les contraintes} \quad \sum_{i=1}^{n} a_{i,p+1} \leq W \tag{2.24}$$

$$b_{p+1} = m - \sum_{k=1}^{m} r_k \tag{2.25}$$

$$r_k \leq 1 \qquad \forall k \in \{1, \ldots, m\} \tag{2.26}$$

$$r_k \leq \sum_{i \in C_k} a_{i,p+1} \qquad \forall k \in \{1, \ldots, m\} \tag{2.27}$$

$$a_{i,p+1} \in \{0, 1\} \qquad \forall i \in \{1, \ldots, n\} \tag{2.28}$$

La fonction objectif de AUX_{MCBB} vise à minimiser le coût réduit de la variable t_{p+1} du problème maître MAS_{MCBB}, qui sera associée à la couverture S_{p+1} en cours de construction. Si la valeur optimale de la fonction objectif est nulle, alors la solution courante de MAS_{MCBB} est optimale pour MCBB ; sinon MAS_{MCBB} doit être résolu à nouveau après l'ajout de la nouvelle couverture.

La contrainte (2.24) assure le respect de la contrainte de bande passante. (2.25) définit b_{p+1} comme le défaut de couverture de S_{p+1}, c'est-à-dire le nombre de cibles non couvertes. Les contraintes qui suivent assurent que r_k est inférieure ou égale à un, et prend la valeur zéro si aucun capteur de S_{p+1} ne couvre la cible k. Bien que r_k doive être à valeurs binaires pour tout k dans $\{1, \ldots, m\}$, il n'est pas nécessaire d'imposer cette contrainte explicitement car l'intégralité des variables $a_{i,p+1}$ implique celle de r_k.

Algorithme génétique pour la résolution du problème auxiliaire L'algorithme génétique GA constitue une alternative efficace à AUX_{MCBB} pour produire des couvertures profitables. Tant que de telles couvertures peuvent être produites par GA, cet algorithme est utilisé avec le critère d'arrêt suivant : GA se termine si la meilleure solution trouvée n'a pas été améliorée au cours des N dernières itérations. En cas d'échec à retourner une couverture profitable, il est relancé jusqu'à trois fois avec un critère d'arrêt basé sur $N' > N$. Ainsi, AUX_{MCBB} n'est résolu que pour échapper à un optimum local, ou pour prouver qu'aucune couverture profitable ne peut être trouvée. L'objectif est donc de minimiser le nombre de fois où AUX_{MCBB} est résolu, ce qui permet de réduire considérablement le temps de calcul (voir section 2.2.4). De plus, GA peut retourner jusqu'à $MAXCOVERS$ couvertures profitables à chaque itération de l'algorithme de génération de colonnes, ce qui a pour effet d'accélérer encore plus sensiblement la convergence.

Les caractéristiques principales de GA sont les suivantes :

Représentation d'un chromosome : un chromosome est représenté par un vecteur de n valeurs binaires. Le $i^{\text{ème}}$ coefficient vaut un si et seulement si le capteur i appartient à la couverture considérée.

Fonctions d'évaluation : trois fonctions d'évaluation sont utilisées. La fonction d'évaluation primaire est la fonction objectif, c'est-à-dire le coût réduit associé à la couverture recherchée, dans le problème maître. La fonction d'évaluation secondaire est le défaut de couverture de la couverture, et la fonction d'évaluation tertiaire est le nombre de capteurs constituant la couverture. La fonction d'évaluation secondaire permet de départager deux couvertures en cas d'égalité sur la fonction d'évaluation primaire, la troisième fonction permettant de trancher en cas d'égalité sur les deux premières. Ces trois fonctions sont utiles car il n'est pas rare que de nombreuses couvertures aient la même valeur pour les fonctions d'évaluation primaires et secondaires.

Sélection : elle s'appuie sur le tournoi binaire, où le meilleur élément de la population est sélectionné avec la probabilité p_{sel}, pour le croisement.

Croisement : l'opérateur de croisement utilisé ici consiste à effectuer un OU logique entre les vecteurs binaires représentant chacun un parent sélectionné. Cela revient à créer une nouvelle solution (un enfant) en faisant l'union des capteurs de ses deux parents.

Mutation : après le croisement, l'enfant est altéré par la mutation comme suit. On applique un NON logique à chaque élément du vecteur binaire représentant l'enfant avec une probabilité p_m.

Opérateur de réparation : la solution obtenue suite au croisement et à la mutation peut être une couverture non valide car le nombre de capteurs peut être strictement supérieur à W. La première phase de l'opérateur

de réparation vise à transformer cette solution en une couverture valide, et la seconde phase a pour objectif de l'améliorer en tentant de retirer des capteurs de la couverture. A chaque itération de la première phase, on calcule G_i, le nombre de cibles qui ne sont couvertes que par le capteur i dans la solution courante. Le capteur associé à la plus petite valeur $G_i + y_i$ est alors retiré de la solution. Ce processus se répète tant que la couverture courante est constituée de plus de W capteurs. Le choix du capteur à supprimer se justifie comme suit : en premier lieu, le retrait d'un capteur pour lequel G_i est petit conduira à une augmentation minimale du défaut de couverture de la solution. En second lieu, le retrait d'un capteur pour lequel y_i est petit aura un impact plus limité sur la dégradation de la valeur de l'objectif.

La seconde phase est similaire à la première, à la différence suivante près. On trie les capteurs par $G_i + y_i$ croissants, et on ne retire le capteur i que si la quantité $G_i + y_i$ est strictement négative.

Stratégie de remplacement : GA repose sur une stratégie de remplacement stationnaire de la population [19]. Dans cette méthode, le dernier enfant produit remplace le plus mauvais élément de la population. Cela permet notamment d'éviter que plusieurs solutions identiques ne coexistent dans la population. Dans GA, on s'assure que tout enfant créé n'existe pas déjà dans la population, on supprime le plus mauvais élément de cette population et on ajoute l'enfant, quelle que soit sa qualité.

Génération de la population initiale : deux algorithmes sont disponibles pour générer la population initiale, ils sont appelés avec la même probabilité. Le premier algorithme consiste à créer des couvertures en choisissant W capteurs au hasard. Le second algorithme est itératif. Les deux premiers capteurs de chaque couverture sont choisis comme suit. Le premier est tiré au hasard parmi les capteurs i satisfaisant $y_i < 0$, le second est tiré au hasard sans condition. Le processus itératif commence alors. A chaque itération, on calcule D_i, le nombre de cibles couvertes par le capteur i (qui n'appartient pas à la couverture en cours de construction), qui ne sont couvertes par aucun capteur de la couverture en construction. Ensuite, on ajoute à la couverture le capteur i pour lequel la quantité $D_i + y_i$ est maximale, à condition qu'elle soit strictement positive. Le processus se termine lorsqu' il n'existe plus aucun capteur vérifiant $D_i + y_i > 0$, lorsque la couverture couvre toutes les cibles, ou lorsque la couverture compte W capteurs.

On vérifie que chaque nouvelle couverture ainsi créée n'existe pas déjà dans la population avant de l'y ajouter. Si cette solution existe déjà elle est écartée et une autre solution est créée.

2.2.4 Résultats numériques

L'ordinateur employé utilise un processeur Intel Pentium IV cadencé à 3.2 GHz muni de 1 gigaoctet de RAM sous Microsoft Windows XP, le solver est GLPK et les algorithmes sont codés en C.

Paramètres de l'algorithme génétique

La taille de la population de GA est de $\min(n, 200)$. On fixe $N = 500$ et $N' = 2500$. GA se termine s'il ne parvient pas à créer une solution n'existant pas déjà dans la population après vingt tentatives infructueuses consécutives. Deux valeurs différentes de p_{sel} (voir le paragraphe consacré à la sélection dans la section 2.2.3) sont utilisées dans GA pour sélectionner les deux parents. Le premier parent est sélectionné en choisissant $p_{sel} = 0.9$, le second avec $p_{sel} = 0.8$. Concernant la mutation (voir le paragraphe correspondant de la section 2.2.3), on fixe $p_m = 0.05$. La valeur de ces paramètres a été choisie empiriquement, elle reste la même pour toutes les instances.

Instances utilisées

Quatre classes d'instances sont considérées : chacune d'elles correspond à un couple de valeurs numériques pour le nombre de capteurs et le nombre de cibles. Comme dans [69], les capteurs et les cibles sont générés aléatoirement dans une zone carrée de 500×500 (l'unité de longueur est le mètre, on aurait pu choisir une autre unité sans perte de généralité). Chaque classe d'instances est divisée en trois sous-classes, en fonction du nombre maximal de capteurs par couverture, W (afin de satisfaire les contraintes de bande passante). Les trois valeurs de W sont $\{5, 10, n\}$. Le cas $W = n$ correspond à l'absence de contraintes de bande passante. Il faut également choisir une durée de vie minimale T_0 pour le réseau. On fixe la valeur de ce paramètre comme dans [69], à l'aide de la formule suivante :

$$T_0 = \left\lfloor \frac{n}{W} \right\rfloor$$

Dans le cas $W = n$, T_0 est fixé à $\lfloor \frac{n}{10} \rfloor$. Le rayon de couverture de chaque capteur est de 150 mètres pour tous les capteurs, comme dans [69].

Chaque classe d'instances est constituée de trente instances, toute sous-classe compte dix instances. Les résultats sont reportés dans la table 2.1. GLPK désigne l'algorithme de génération de colonnes pour lequel le problème

auxiliaire est résolu par GLPK. GA_1 fait référence à l'algorithme de génération de colonnes pour lequel le problème auxiliaire est résolu avec GA comme indiqué par le logigramme de la figure 2.3 avec $MAXCOVERS = 1$ (c'est-à-dire qu'un seule colonne est retournée par GA à chaque itération). GA_{20} désigne le même algorithme, mais pour lequel jusqu'à $MAXCOVERS = 20$ couvertures profitables sont retournées par GA à chaque itération. La table ne montre pas les valeurs atteintes par la fonction objectif, car ces valeurs sont identiques pour les trois algorithmes (qui sont tous les trois des approches exactes). La comparaison ne peut donc s'opérer que sur le temps de calcul. Les trois colonnes correspondantes dans la table 2.1 montrent le temps de calcul moyen de chaque algorithme sur les dix instances, exprimé en secondes. Les trois colonnes suivantes montrent le nombre moyen d'itérations de l'algorithme de génération de colonnes, et les trois dernières colonnes de la table font apparaître le nombre moyen de colonnes générées avant d'obtenir une solution optimale.

TABLE 2.1 – Résultats pour MCBB.

				Temps de calcul			Nb. Itérations			Nb. Couvertures		
n	m	W	T_0	GLPK	GA_1	GA_{20}	GLPK	GA_1	GA_{20}	GLPK	GA_1	GA_{20}
50	30	50	5	0.23	0.33	0.25	34.30	28.57	11.17	33.30	27.57	203.33
		10	5	0.23	0.35	0.26	33.67	28.30	11.13	32.67	27.30	202.67
		5	10	0.68	0.84	0.61	85.40	83.70	24.53	84.40	82.70	470.13
100	60	100	10	3.97	1.79	0.75	115.27	69.83	17.63	114.27	68.83	332.67
		10	10	3.87	1.56	0.78	111.10	67.83	16.43	110.10	66.83	308.67
		5	20	16.42	5.79	2.90	234.67	240.13	53.00	233.67	239.13	1035.10
150	90	150	15	22.83	6.86	2.13	235.17	135.07	26.10	234.17	134.07	502.00
		10	15	20.53	4.35	1.98	202.87	119.97	25.27	201.87	118.97	485.33
		5	30	447.10	61.30	45.61	429.37	496.50	96.90	428.37	495.50	1635.43
200	120	200	20	98.63	18.05	4.44	391.97	200.17	32.33	390.97	199.17	626.67
		10	20	66.73	10.20	3.54	301.00	176.67	30.43	300.00	175.67	588.67
		5	40	2266.78	181.88	140.32	635.00	846.60	168.93	634.00	845.60	2365.33

Lorsque le problème auxiliaire est résolu avec GLPK, une couverture est produite à chaque itération (exception faite de la dernière itération qui ne sert qu'à prouver l'optimalité de la solution courante du problème maître en démontrant qu'il n'existe plus de couverture profitable).

Lorsque W diminue, le nombre de couvertures valides (et donc l'espace des solutions) diminue également. Cependant, on observe que quel que soit l'algorithme utilisé, le problème devient nettement plus difficile lorsque les contraintes de bande passante sont fortes (c'est-à-dire lorsque W diminue).

La table 2.1 montre que la résolution de la formulation basée sur la programmation linéaire en nombres entiers du problème auxiliaire à chaque itération n'est efficace que pour des instances de petite taille. L'approche combinant GA et cette formulation s'avère beaucoup plus rapide lorsque la taille des instances augmente : le temps de calcul peut être jusqu'à dix fois plus faible pour $n = 200$, $m = 120$ et $W = 5$.

La génération de plusieurs colonnes profitables à chaque itération est particulièrement efficace pour le temps de calcul comme pour le nombre d'itérations. Ce résultat est très intéressant car il montre que générer la meilleure colonne (au sens du coût réduit) à chaque itération ne conduit pas à minimiser le nombre d'itérations. En effet, un ensemble de colonnes profitables offre davantage de possibilités au problème maître pour améliorer la valeur de l'objectif que la génération d'une seule colonne, fût-elle celle qui minimise le coût réduit. L'aptitude de GA à retourner rapidement un ensemble de colonnes profitables est dûe aux fonctions d'évaluation utilisées et à la fonction de réparation qui s'efforce d'éliminer les capteurs redondants de toute nouvelle solution, alors que AUX_{MCBB} se focalise sur le seul coût réduit, au détriment des autres caractéristiques désirables des couvertures.

On pourrait objecter que l'efficacité de l'approche hybride proposée n'est dûe qu'aux mauvaises performances du solver utilisé (GLPK en l'occurrence), et que l'emploi d'un solver commercial pour résoudre AUX_{MCBB} pourrait relativiser l'intérêt de GA. En effet, les solvers commerciaux se montrent généralement plus performants que leurs homologues gratuits car ils bénéficient de stratégies plus sophistiquées pour produire et gérer des coupes, pour brancher, produire des bornes inférieures et obtenir des solutions faisables. Ces caractéristiques sont à l'origine de l'écart de performances entre solvers gratuits et commerciaux, qui est d'autant plus manifeste que les problèmes à résoudre sont difficiles et de grande taille. Cependant, l'écart de performances pour un problème de taille plus modeste, est parfois assez marginal. C'est typiquement le cas de AUX_{MCBB}, dont la solution reste calculable en quelques minutes avec GLPK. En effet, l'implémentation de l'approche proposée avec XPRESS-MP [72] n'est que 20% plus rapide que la version basée sur GLPK (et jusqu'à 50% plus rapide pour $W = 5$). Par conséquent l'approche hybride proposée (c'est-à-dire la combinaison de GA et de GLPK pour résoudre le problème auxiliaire) reste sensiblement plus efficace que l'approche basée sur la résolution du problème auxiliaire par la seule formulation

par programmation linéaire en nombres entiers, même avec un solver commercial. Le choix du solver s'est porté sur GLPK en raison des restrictions imposées par la licence d'utilisation de XPRESS-MP, qui n'était pas disponible à l'Université d'Hyderabad où ce travail a été développé (le programme *Academic Initiative* d'IBM ne concernait pas encore ILOG-CPLEX au moment où ces travaux ont commencé).

2.3 Maximisation de la durée de vie du réseau (MNLB)

2.3.1 Définition du problème

Cette section est dédiée au problème de maximisation de la durée de vie du réseau sous contrainte de bande passante, et de qualité de service (la qualité de service est mesurée en termes de défaut de couverture global), noté MNLB. Ce problème est très proche de MCBB :

- La contrainte de durée de vie minimale de MCBB devient l'objectif de MNLB (à maximiser)
- L'objectif de minimisation du défaut de couverture de MCBB devient une contrainte dans MNLB

2.3.2 Algorithme de génération de colonnes

L'approche proposée est identique à celle qui a été présentée pour aborder MCBB sur la figure 2.3 : le problème maître noté MAS_{MNLB} ordonnance les couvertures S_j pour tout j dans $\{1, \ldots, p\}$ afin de maximiser la durée de vie du réseau notée LT (pour *lifetime*) tout en respectant une qualité de service minimale. Cette qualité de service s'exprime comme un défaut de couverture global maximal à ne pas dépasser. Le problème auxiliaire consiste à construire des couvertures profitables satisfaisant la contrainte de bande passante, afin de permettre au problème maître d'augmenter la durée de vie courante.

Dans le problème maître, on s'assure que le défaut de couverture global normalisé noté BR (pour *breach rate*) reste inférieur à α, où α est un réel donné dans l'intervalle $[0, 1]$. Le défaut de couverture global normalisé est défini par :

$$BR = \frac{\sum_{j=1}^{p} b_j t_j}{m \sum_{j=1}^{p} t_j}$$

Comme $0 \leq b_j \leq m$ pour tout j, on vérifie que BR est bien dans l'intervalle $[0, 1]$. $BR = 0$ implique que toutes les cibles sont couvertes au cours de l'intervalle $[0, LT]$; $BR = 1$ signifie qu'aucune cible n'est jamais couverte. $BR = 0.1$ signifie que 90% des cibles sont couvertes dans l'intervalle $[0, LT]$. Cependant, cette même mesure peut recouvrir toute situation intermédiaire entre les deux scénarios extrêmes suivants :

- 90% des cibles sont toujours couvertes pendant qu'aucune des cibles restantes ne l'est jamais
- La durée pendant laquelle chaque cible est couverte est égale à $0.9 \times LT$

2.3.3 Problème maître

La contrainte sur le défaut de couverture global peut s'écrire $BR \leq \alpha$. Par définition de BR, on a :

$$\sum_{j=1}^{p} \Big(b_j - m\alpha \Big) t_j \leq 0$$

Le problème maître pour MNLB, noté MAS_{MNLB}, est alors :

$$\text{Maximiser} \quad \sum_{j=1}^{p} t_j \tag{2.29}$$

$$\text{Sous les contraintes} \quad \sum_{j=1}^{p} a_{i,j} t_j \leq 1 \qquad \forall i \in \{1,\ldots,n\} \qquad [y_i] \tag{2.30}$$

$$\sum_{j=1}^{p} \Big(b_j - m\alpha\Big) t_j \leq 0 \qquad [y_{n+1}] \tag{2.31}$$

$$t_j \geq 0 \qquad \forall j \in \{1,\ldots,p\} \tag{2.32}$$

La durée de vie du réseau de capteurs constitue maintenant la fonction objectif du problème. Comme MAS_{MNLB} est un problème de maximisation, la couverture profitable S_{p+1} est associée à la nouvelle variable non basique t_{p+1} de MAS_{MNLB}, dont le coût réduit doit être strictement positif.

La nouvelle couverture S_{p+1} à déterminer par le problème auxiliaire est représentée par les n premiers éléments du vecteur colonne $A_{p+1} \in \mathbb{R}^{n+1}$:

$$A_{p+1} = \begin{bmatrix} a_{1,p+1} \\ a_{2,p+1} \\ \vdots \\ a_{n,p+1} \\ b_{p+1} - m\alpha \end{bmatrix}$$

Le dernier élément de A_{p+1} correspond à la contrainte sur le défaut de couverture global.

2.3.4 Problème auxiliaire

Le problème auxiliaire pour MNLB consiste à déterminer une couverture profitable S_{p+1} pour le problème maître, c'est-à-dire calculer $a_{i,p+1} \in \{0,1\}$ pour tout $i \in \{1,\ldots,n\}$ tout en satisfaisant la contrainte de bande passante, et en maximisant le coût réduit de la variable correspondante t_{p+1} qui lui sera associée dans le problème maître. Comme dans le cas de MCBB, le problème auxiliaire est d'abord écrit comme un programme linéaire en nombres entiers, puis un algorithme génétique appelé GA est présenté pour produire des couvertures profitables.

Formulation du problème auxiliaire sous la forme d'un programme linéaire en nombres entiers Cette formulation est notée AUX_{MNLB}. Sa fonction objectif vise à maximiser le coût réduit de la nouvelle couverture, qui est $1 - y^\top A_{p+1}$. A l'exception de sa fonction objectif, AUX_{MNLB} est identique à AUX_{MCBB}.

$$\text{Maximiser} \quad 1 - \sum_{i=1}^{n} y_i a_{i,p+1} - y_{n+1}\Big(b_{p+1} - m\alpha\Big) \tag{2.33}$$

$$\text{Sous les contraintes} \quad \sum_{i=1}^{n} a_{i,p+1} \leq W \tag{2.34}$$

$$b_{p+1} = m - \sum_{k=1}^{m} r_k \tag{2.35}$$

$$r_k \geq a_{i,p+1} \qquad \forall k \in \{1,\ldots,m\} \forall i \in C_k \tag{2.36}$$

$$r_k \leq 1 \qquad \forall k \in \{1,\ldots,m\} \tag{2.37}$$

$$r_k \leq \sum_{i \in C_k} a_{i,p+1} \qquad \forall k \in \{1,\ldots,m\} \tag{2.38}$$

$$a_{i,p+1} \in \{0,1\} \qquad \forall i \in \{1,\ldots,n\} \tag{2.39}$$

Algorithme génétique pour la résolution du problème auxiliaire

L'algorithme génétique utilisé pour MNLB est très proche de celui qui a été développé pour MCBB. On ne mentionne ici que ce qui le distingue de la version présentée à la section 2.2.3.

GA ne repose que sur deux fonctions d'évaluation. La fonction d'évaluation primaire est la fonction objectif de MNLB, et la fonction d'évaluation secondaire est le nombre de capteurs dans la couverture. Là aussi, la fonction d'évaluation secondaire permet d'opérer un arbitrage lorsque deux couvertures ont la même valeur pour la fonction d'évaluation primaire. Dans la nouvelle fonction de réparation, la première phase retire le capteur i pour lequel $y_i - y_{n+1}G_i$ est maximum. Dans la seconde phase de la fonction de réparation, on retire désormais le capteur i pour lequel $y_i - y_{n+1}G_i$ est maximum, à condition que cette quantité soit strictement positive. Cette seconde phase prend fin lorsque la valeur maximale de $y_i - y_{n+1}G_i$ est négative. Le second algorithme de génération de la population initiale utilise l'une ou l'autre des deux méthodes suivantes pour choisir les deux premiers capteurs, avec la même probabilité. La première méthode consiste à les choisir au hasard parmi ceux pour lesquels $y_i > 0$, la seconde méthode choisit deux capteurs au hasard dans l'ensemble $\{1, \ldots, n\}$. Ensuite, on ajoute le capteur pour lequel la quantité $y_{n+1}D_i - y_i$ est maximale à condition qu'elle soit strictement positive. L'algorithme se termine s'il n'existe plus aucun capteur pour lequel $y_{n+1}D_i - y_i > 0$, si la couverture couvre toutes les cibles, ou si elle compte W capteurs.

2.3.5 Résultats numériques

Les instances considérées sont identiques à celles utilisées pour MCBB à ceci près qu'à la différence de MCBB, MNLB requiert la donnée du défaut de couverture global normalisé α. On a considéré trois valeurs pour ce paramètre : $\alpha = 0$ signifie que toutes les cibles doivent être couvertes pendant toute la durée de vie du réseau, $\alpha = 0.1$ implique une couverture globale d'au moins 90% et $\alpha = 0.2$ garantit une couverture globale supérieure à 80%. Tous les paramètres de l'algorithme génétique sont réglés de la même manière que pour MCBB.

Les résultats obtenus pour les trois valeurs de α sont reportés dans les tables 2.2, 2.3 et 2.4 respectivement.

TABLE 2.2 – Résultats pour MNLB avec $\alpha = 0$.

			Temps de calcul			Nb. Itérations			Nb. Couvertures		
n	m	W	GLPK	GA_1	GA_{20}	GLPK	GA_1	GA_{20}	GLPK	GA_1	GA_{20}
		50	0.18	0.23	0.14	22.03	19.20	6.60	21.03	18.20	112.00
50	30	10	0.18	0.30	0.17	22.80	19.27	7.03	21.80	18.27	120.67
		5	0.16	0.35	0.27	15.00	16.37	5.87	14.00	17.37	93.80
		100	2.47	1.40	0.45	63.93	52.37	10.07	62.93	51.37	181.33
100	60	10	2.53	1.32	0.59	62.13	49.10	10.07	61.13	48.10	181.33
		5	1.42	1.10	0.85	14.83	16.20	6.23	13.83	15.20	90.20
		150	11.61	4.21	1.03	104.17	76.83	12.23	103.17	75.83	224.67
150	90	10	11.80	3.11	1.13	96.67	70.17	12.17	95.67	69.17	223.33
		5	5.47	2.76	2.33	12.77	15.70	6.70	11.77	14.70	75.03
		200	39.78	11.14	2.24	167.10	118.17	16.37	166.10	117.17	307.33
200	120	10	39.02	7.03	2.00	141.27	105.00	15.30	142.27	104.00	286.00
		5	9.62	4.59	3.79	11.63	16.23	7.13	10.63	15.23	69.63

TABLE 2.3 – Résultats pour MNLB avec $\alpha = 0.1$.

			Temps de calcul			Nb. Itérations			Nb. Couvertures		
n	m	W	GLPK	GA_1	GA_{20}	GLPK	GA_1	GA_{20}	GLPK	GA_1	GA_{20}
		50	0.83	0.93	0.62	106.17	103.80	30.40	105.17	102.80	588.00
50	30	10	0.83	1.05	0.74	105.27	109.90	32.17	104.27	108.90	623.33
		5	0.64	0.63	0.45	73.93	73.90	18.20	72.93	72.90	342.93
		100	19.24	10.60	5.15	467.53	412.60	84.63	466.53	411.60	1670.9
100	60	10	19.95	10.51	4.66	465.17	426.07	79.83	464.17	425.07	1573.00
		5	23.67	7.00	2.81	225.60	238.57	49.43	224.60	237.57	937.33
		150	313.48	57.69	29.92	945.20	831.17	148.30	944.20	830.17	2772.27
150	90	10	325.83	51.90	37.00	925.50	844.63	147.27	924.50	843.63	2716.07
		5	551.96	53.08	38.44	440.00	509.17	98.87	439.00	508.17	1530.83
		200	1117.20	147.36	62.10	1459.63	1279.67	209.50	1458.63	1278.67	3881.60
200	120	10	1026.12	138.32	60.04	1385.43	1310.37	216.93	1384.43	1309.37	3824.07
		5	2813.32	243.67	221.97	651.20	857.20	190.67	650.20	856.20	2316.03

On observe d'abord que la difficulté du problème augmente avec α. En effet, le nombre de couvertures permettant de satisfaire la contrainte de qualité de service augmente avec α. Comme dans le cas de MCBB, l'efficacité apportée

TABLE 2.4 – Résultats pour MNLB avec $\alpha = 0.2$.

			Temps de calcul			Nb. Itérations			Nb. Couvertures		
n	m	W	GLPK	GA_1	GA_{20}	GLPK	GA_1	GA_{20}	GLPK	GA_1	GA_{20}
50	30	5	0.85	0.67	0.50	113.10	115.37	25.37	112.10	114.37	485.63
		10	1.09	1.13	0.82	152.37	161.03	38.90	151.37	160.03	756.93
		50	1.07	1.08	0.78	152.40	153.20	38.63	151.40	152.20	751.73
100	60	5	25.21	8.78	2.95	331.50	343.03	59.67	330.50	342.03	1156.90
		10	30.33	11.42	5.45	565.80	532.57	91.93	564.80	531.57	1803.10
		100	30.17	11.95	5.71	565.13	516.13	93.93	564.13	515.13	1843.40
150	90	5	334.24	48.81	26.40	585.03	655.70	113.63	584.03	654.70	1928.63
		10	344.19	65.15	32.13	1052.80	1008.63	169.03	1051.80	1007.63	3045.13
		150	344.48	65.96	34.63	1069.60	982.10	167.03	1068.60	981.10	3137.30
200	120	5	3247.60	287.89	245.65	922.10	1114.97	228.07	921.10	1113.97	2760.47
		10	3235.12	327.06	241.14	1615.30	1664.23	306.20	1614.30	1663.23	4380.83
		200	3196.82	263.65	218.39	1679.70	1592.77	276.87	1678.70	1591.77	4484.83

par GA se révèle lorsque α et la taille des instances augmentent. Cette approche est efficace parce qu'elle évite de recourir à la résolution d'un programme linéaire en nombres entiers pour produire des couvertures profitables, et la réduction du temps de calcul reste significative même lorsque cette stratégie conduit à générer un très grand nombre total de colonnes. Comme pour MCBB, l'approche proposée est environ dix fois plus rapide sur les instances de grande taille ($n = 200$ et $m = 120$) que l'approche pour laquelle le problème auxiliaire est résolu par le solver à chaque itération.

2.3.6 Heuristique obtenue à partir de l'approche proposée

Comme les réseaux de capteurs sans fil sont parfois constitués de dizaines de milliers de nœuds [73], des heuristiques peuvent être utiles lorsque les approches exactes ne permettent plus d'obtenir de solutions optimales en temps raisonnable. Les résultats des sections 2.2.4 et 2.3.5 font apparaître que MNLB est le problème le plus difficile, en particulier lorsque $\alpha = 0.2$. C'est ce problème qui est abordé ici à l'aide d'une heuristique identique à l'approche hybride proposée, pour pour laquelle la formulation par programmation linéaire en nombres entiers du problème auxiliaire n'est jamais résolue. Lorsque GA ne parvient plus à trouver de couvertures profitables après quatre essais, l'algorithme se termine.

Les résultats de cette heuristique sont donnés dans la table 2.5. La première colonne rapporte le temps de calcul moyen de l'heuristique, en secondes. La seconde colonne rappelle le temps de calcul de l'approche exacte utilisant GA_{20} pour résoudre le problème auxiliaire, et GLPK quand GA_{20} ne trouve plus de couverture profitable (cette colonne est identique à la troisième colonne de la table 2.4). La troisième colonne de la table 2.5 rapporte le gain de temps en pourcentage, par rapport à l'approche exacte. Enfin, la dernière colonne montre la dégradation moyenne de la durée de vie de la solution retournée par l'heuristique, comparée à la durée de vie optimale, exprimée en pourcentage. Lorsque cette dégradation moyenne vaut 0%, cela signifie que GA a été capable de générer toutes les colonnes utiles à l'obtention d'une solution optimale, mais sans que l'optimalité ne puisse être prouvée.

On peut voir que l'heuristique permet de réduire le temps de calcul jusqu'à 90% pour les instances les plus difficiles. Le fait que le gain de temps fourni par l'heuristique augmente avec la difficulté des instances est une caractéristique désirable pour une heuristique. Naturellement, on observe aussi une augmentation de la dégradation des solutions produites avec la taille des instances, mais la dégradation est particulièrement faible. Cela s'explique par l'effet plateau de la génération de colonnes. La fin de la recherche avec l'approche exacte demande de nombreuses itérations et produit une augmentation de la durée de vie très marginale. Ces résultats illustrent encore une fois l'efficacité de l'algorithme génétique pour la résolution du problème auxiliaire.

Si les résultats de la table 2.5 sont prometteurs, les temps de calcul de cette heuristique deviennent rapidement prohibitifs comme le montre la table 2.6. Pour $n = 600$ capteurs, le temps de calcul moyen de l'heuristique atteint quasiment une heure. Cela peut s'expliquer par la combinaison des causes suivantes :

- Alors que la taille de l'espace des solutions augmente avec n (et W) le nombre d'itérations requis pour obtenir une bonne solution ne peut lui aussi qu'augmenter
- Le nombre de colonnes du problème maître devient très grand avec l'augmentation du nombre d'itérations, ce qui tend à rendre sa résolution de plus en plus longue
- En raison de l'effet plateau [67], GA doit produire un très grand nombre de couvertures pendant de nombreuses itérations, alors même que la durée de vie de la solution optimale du problème maître n'augmente que très marginalement.

TABLE 2.5 – Comparaison de l'heuristique et de l'approche exacte dont elle dérive pour MNLB avec $\alpha = 0.2$.

n	m	W	Temps CPU Heuristique	Temps CPU Appr. exacte	Réduction du temps CPU	Dégradation de la durée de vie
		50	0.77	0.78	1.27%	0.00%
50	30	10	0.79	0.82	4.30%	0.00%
		5	0.49	0.50	3.23%	0.00%
		100	5.26	5.71	7.98%	0.00%
100	60	10	5.07	5.45	6.97%	0.00%
		5	2.63	2.95	10.91%	0.00%
		150	18.89	34.63	45.46%	0.01%
150	90	10	16.80	32.13	47.73%	0.01%
		5	8.64	26.40	67.26%	0.02%
		200	48.51	218.39	77.79%	0.02%
200	120	10	43.81	241.14	81.83%	0.03%
		5	23.40	245.65	90.48%	0.03%

TABLE 2.6 – Résultats de l'heuristique pour MNLB avec $\alpha = 0.2$ pour des instances de grande taille.

n	m	W	Temps de calcul	Nb. Itérations	Nb. Couvertures
		600	3470.62	1593.30	19454.40
600	450	10	1866.95	1374.10	15859.30
		5	814.21	998.20	8537.30

Le principal défaut de cette heuristique est que l'algorithme génétique a été initialement conçu pour une approche exacte, ce qui conduit à un temps de calcul trop élevé car aucun compromis n'est fait sur la qualité de la solution. Un rééquilibrage en faveur du temps de calcul devrait être opéré. Cela peut se faire en limitant le nombre de tentatives de l'algorithme génétique (qui peut aller jusqu'à quatre dans l'heuristique actuelle). En outre, un mécanisme permettant de purger le problème maître des couvertures générées lors des toutes premières itérations, et qui n'ont plus été utilisées depuis longtemps, conduirait sans doute à conserver un temps de calcul très faible pour le problème maître.

2.4 Exploration de compromis entre qualité de service et durée de vie

MCBB et MNLB requièrent tous les deux la fixation d'un paramètre : la durée de vie minimale du réseau de capteurs T_0 pour MCBB, et le défaut de couverture global normalisé α pour MNLB. Bien que la durée de vie du réseau soit comprise dans l'intervalle $[1, n]$ et que le défaut de couverture global normalisé soit dans l'intervalle $[0, 1]$, fixer ces paramètres *a priori* reste difficile car la distribution spatiale des capteurs et des cibles a un impact très fort sur la durée de vie et la qualité de service que l'on peut raisonnablement attendre du réseau de capteurs. Cette section vise à fournir un aperçu des performances que l'on peut attendre d'un réseau, ce qui permet de fixer T_0 ou α en connaissance de cause. L'intérêt de la réutilisation des couvertures est également mis en évidence.

2.4.1 Définition du problème et approche proposée

Le problème consistant à explorer les compromis entre qualité de service et durée de vie est noté TOLB pour *trade-offs between lifetime and total breach*. On se propose ici de déterminer un ensemble de couples $(T_0, 1-\alpha)$ (où $1-\alpha$ est la couverture globale normalisée) représentant des compromis. Comme la couverture globale normalisée et la durée de vie du réseau sont des critères antagonistes, TOLB s'attache à couvrir un large spectre de compromis possibles.

On s'intéresse ici à MCBB. On peut bien sûr envisager de résoudre TOLB en résolvant MCBB pour un ensemble de valeurs de T_0 préalablement définies. On résoudra MCBB F fois (avec $F \geq 2$), mais de manière à ce que les couvertures produites pour une valeur de T_0, et qui sont utilisées dans la solution optimale, puissent constituer la solution initiale pour la prochaine résolution de MCBB où T_0 prendra une valeur inférieure. Cela implique de commencer par $T_0 = n$ et de la décrémenter itérativement jusqu'à 1. En effet, la durée de vie d'un capteur est d'une unité de temps, aussi la durée de vie du réseau ne peut pas être inférieure à un (même si on peut considérer qu'elle est nulle lorsque MCBB n'admet pas de solution). Ainsi, les couvertures constituant la solution optimale pour une valeur de T_0 forment une solution admissible pour la prochaine valeur de T_0. Plus la valeur suivante de T_0 est proche de la valeur courante, meilleure est la solution initiale ainsi produite pour le problème suivant. Les résultats

TABLE 2.7 – Résultats détaillés de TOLB pour une instance caractérisée par $n = 200$, $m = 120$ et $W = 5$.

		Réutilisation des couvertures		Pas de réutilisation	
T_0	$1-\alpha$	Temps de calcul	Nb. Couvertures	Temps de calcul	Nb. Couvertures
200.000	0.216	2.19	20	2.16	20
177.889	0.243	2.34	80	2.33	80
155.778	0.278	3.95	460	2.97	240
133.667	0.323	4.36	620	4.45	460
111.556	0.388	4.55	580	6.67	640
89.444	0.483	5.86	820	7.84	1100
67.333	0.642	6.34	1100	11.64	1720
45.222	0.863	33.72	2403	72.47	2729
23.111	0.957	124.06	776	167.87	858
1.000	1.000	1.70	0	1.75	20
Total		**189.08**	**6859**	**280.16**	**7867**

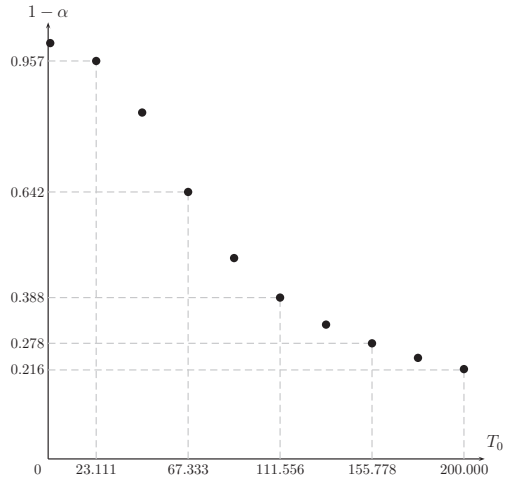


FIGURE 2.4 – Compromis entre couverture globale et durée de vie pour une instance caractérisée par $n = 200$, $m = 120$ et $W = 5$.

numériques de la table 2.8 montrent que cette manière de procéder est sensiblement plus efficace que de résoudre MCBB F fois sans tirer parti des colonnes produites d'un problème à l'autre.

MCBB est résolu à l'aide de la méthode exacte la plus efficace présentée ici, c'est-à-dire celle qui utilise GA_{20} pour résoudre le problème auxiliaire, puis GLPK lorsque GA_{20} ne trouve aucune couverture profitable après quatre tentatives. Le nombre de compromis, F, qui est égal au nombre de fois où MCBB doit être résolu, est fixé par l'utilisateur. On considère alors F valeurs de T_0 également réparties dans l'intervalle $[1, n]$ pour résoudre MCBB, avec $T_0 \leftarrow 1 + (n-1)\dfrac{f}{F-1}$, pour $f \in \{F-1, \ldots, 0\}$. MCBB est donc d'abord résolu pour $T_0 = n$, qui est un problème très facile car l'unique solution optimale consiste à utiliser chaque couverture initiale (constituée d'un seul capteur) pendant une unité de temps.

2.4.2 Résultats numériques

Les résultats obtenus sont d'abord présentés en détail sur une instance contenant $n = 200$ capteurs, $m = 120$ cibles, avec $W = 5$. L'approche proposée et celle qui consiste à résoudre MCBB sans réutiliser les colonnes des problèmes précédents sont comparées en termes de temps de calcul, et de nombre de couvertures produites. On a arbitrairement choisi $F = 10$. La table 2.7 montre que le premier et le dernier problème (correspondant à $T_0 = n$ et $T_0 = 1$ respectivement) sont faciles, et que la réutilisation des couvertures conduit à une réduction du temps de calcul d'environ 30%, et du nombre de couvertures d'environ 12%. Plus précisément, la réutiliusation des couvertures n'accélère pas la convergence quand T_0 est élevé, mais sa diminution progressive rend le problèlme plus difficile et fait clairement apparaître l'effet bénéfique de la réutilisation des couvertures sur le temps de calcul. La figure 2.4 montre les dix paires $(T_0, 1-\alpha)$ de la table 2.7. Cette figure est une approximation du front de Pareto (les solutions pouvant être dominées [65]).

TABLE 2.8 – Résolution de TOLB avec et sans réutilisation des couvertures.

			Réutilisation des couvertures		Pas de réutilisation	
n	m	W	Temps de calcul	Nb. Couvertures	Temps de calcul	Nb. Couvertures
		50	3.24	2220	4.58	3320
50	30	10	3.10	2020	3.94	2940
		5	2.47	1380	2.84	1680
		100	17.71	5764	28.97	7700
100	60	10	16.63	5700	21.52	6980
		5	15.56	4000	17.97	4485
		150	51.78	9500	113.07	15140
150	90	10	41.85	8560	70.42	12660
		5	109.55	6094	136.83	7413
		200	130.45	12708	237.95	19446
200	120	10	113.10	11948	146.45	14953
		5	189.08	6859	280.16	7867

Les résultats de la figure 2.4 font apparaître un autre avantage de l'approche proposée. Si on choisit d'affecter à T_0 une valeur différente de celles qui ont été utilisées par TOLB, alors on peut tirer parti de la résolution de TOLB pour résoudre MCBB en construisant l'ensemble des couvertures initiales comme suit. T_0^- (respectivement T_0^+) est la plus grande valeur (respectivement la plus petite) parmi les F valeurs utilisées par TOLB satisfaisant $T_0^- < T_0$ (respectivement, qui vérifie $T_0 < T_0^+$). Ensuite, on fait l'union des couvertures utilisées dans la solution optimale obtenue pour MCBB avec T_0^- et T_0^+ afin d'en supprimer les éventuels doublons, et l'ensemble de couvertures ainsi obtenu constitue les colonnes initiales du problème maître de MCBB pour T_0. Notons que « faire l'union » ne signifie pas calculer de nouvelles couvertures, mais simplement rassembler des couvertures existantes. Ainsi, la solution initiale de MCBB pour T_0 est au moins de $1 - \alpha^+$, où α^+ est le défaut de couverture global normalisé correspondant à la solution de MCBB obtenue pour T_0^+. En effet, la présence des couvertures provenant de la solution de MCBB associée à T_0^+ garantit une solution faisable pour T_0, alors que les couvertures provenant de la solution de MCBB associée à T_0^- permettent d'obtenir une solution initiale encore meilleure lorsque le problème maître est résolu pour la première fois pour T_0. Comme la réutilisation des couvertures a fait la preuve de son efficacité, la résolution de MCBB à l'aide des couvertures initiales ainsi construites ne peut que contribuer à faire diminuer le temps de calcul.

La réutilisation des couvertures est maintenant testée sur une instance de chaque taille considérée dans ce chapitre comme le montre la table 2.8. La première colonne donne le temps de calcul en secondes, la suivante est le nombre de couvertures nécessaires à l'obtention des $F = 10$ solutions de MNLB. La table 2.8 montre clairement que la résolution de TOLB peut être jusqu'à 50% plus rapide avec la réutilisation des couvertures, que sans réutilisation. Ce constat est d'autant plus vrai que W est grand. Comme MCBB doit être résolu à de nombreuses reprises, l'emploi d'un algorithme génétique efficace pour résoudre le problème auxiliaire est particulièrement utile pour aborder TOLB, qui peut ainsi retourner dix compromis pour une instance en temps raisonnable (moins de quatre minutes).

2.5 Conclusion

Ce chapitre propose un algorithme basé sur la génération de colonnes pour aborder deux problèmes proches relatifs à l'optimisation de l'exploitation des réseaux de capteurs sans fil soumis à une contrainte de bande passante. L'approche permet en outre d'explorer des compromis permettant de fournir un aperçu des performances que peut atteindre le réseau en termes de qualité de service, et de durée de vie. Les résultats numériques montrent que les algorithmes proposés sont utilisables sur des instances utilisant jusqu'à 200 capteurs, et que des heuristiques peuvent en être déduites moyennant cependant certaines adaptations. Plus généralement, cette contribution illustre le potentiel important offert par la combinaison des métaheuristiques et des méthodes exactes pour résoudre le problème auxiliaire dans un algorithme basé sur la génération de colonnes. Les résultats obtenus montrent que les métaheuristiques permettent assez facilement de prendre en compte certains aspects propres aux problèmes abordés (c'est-à-dire ne se limitant pas au coût réduit et aux contraintes « dures »), et de produire des solutions particulièrement efficaces. Cela suggère que les métaheuristiques ne devraient pas être considérées seulement comme un moyen de résoudre un problème de manière approchée à moindre coût computationnel. En effet, on a observé que le nombre d'itérations pouvait être inférieur à celui de l'algorithme de génération de colonnes n'exploitant que la programmation linéaire en nombres entiers pour résoudre le problème auxiliaire, lorsque les colonnes sont construites intelligemment par l'algorithme génétique. Ce travail ouvre des perspectives au plan des applications (les algorithmes présentés ici peuvent faire l'objet d'adaptations à des variantes des problèmes d'optimisation de

l'exploitation des réseaux sans fil), mais aussi au plan des méthodes, afin de combiner efficacement métaheuristiques et méthodes exactes.

Chapitre 3

Configuration robuste de réseaux de distribution d'électricité

Résumé

Ce chapitre est consacré à la conception d'une configuration pour un réseau de distribution d'électricité basse tension. L'objectif visé est de maximiser la marge de puissance du fournisseur d'électricité le plus chargé. En plus de la contrainte de capacité des fournisseurs (en puissance), la longueur de la chaîne la plus courte (en nombre d'arêtes) entre tout consommateur et son fournisseur doit être inférieure à une borne donnée afin de limiter les pertes de puissance par effet Joule et les chutes de tension. Les conditions de faisabilité du problème sont mises en évidence et conduisent à des résultats de complexité. Une heuristique gloutonne sans garantie de performance est ensuite proposée ainsi que deux formulations basées sur la programmation linéaire en variables mixtes. Un algorithme de plans coupants reposant sur la génération de deux classes de coupes permettant d'assurer la connexité de la solution et de satisfaire la contrainte de distance est proposée pour résoudre la seconde formulation du problème. Toutes les approches proposées dans ce chapitre sont ensuite comparées sur un ensemble de 190 instances, dont les résultats font l'objet d'une analyse et d'une discussion.

3.1 Introduction

La dérégulation des marchés de l'énergie en Europe a considérablement bouleversé la conception et l'utilisation des réseaux d'électricité. Dans [50], on lit que la demande d'électricité a crût de plus de 10% dans l'Union Européenne en 2009 alors que l'incertitude plane sur la question du renouvellement du parc nucléaire européen depuis l'accident survenu à Fukushima au Japon en 2011. Cela conduit à exploiter les réseaux électriques au plus près de leurs limites comme l'indique [17], et l'augmentation du risque de black out qui en découle (voir [47]) souligne le besoin de flexibilité de ces réseaux. Heureusement, l'introduction d'interrupteurs commandables à distance a permis de conférer un surcroît de souplesse aux réseaux électriques.

Un réseau électrique est constitué d'un réseau de transport haute tension, chargé d'alimenter un réseau basse tension dont le maillage est plus fin. Les fournisseurs d'électricité dont il est question ici sont les transformateurs chargés de faire l'interface entre le réseau haute tension, et le réseau basse et moyenne tension qui alimente les consommateurs. Le terme consommateur désigne ici les habitations, les lieux de travail et plus généralement tout bâtiment alimenté en électricité basse tension. En plus des lignes électriques qui transportent le courant, le réseau basse tension est équipé d'interrupteurs, et configurer le réseau basse tension consiste à choisir l'état (ouvert ou fermé) de chaque interrupteur. La configuration a un impact direct sur certains paramètres physiques comme les pertes Joule, le profil de tension et la charge des transformateurs. Un état de l'art des méthodes disponibles pour configurer un réseau basse tension peut-être trouvé dans [18]. Dans [49], qui présente une revue très complète des heuristiques utilisées dans ce contexte entre 1990 et 2000, on observe que le problème le plus souvent abordé dans la littérature consiste à minimiser les pertes Joule, ce qui nécessite de connaître le courant dans toutes les branches du réseau. Des heuristiques basées sur diverses techniques de simulation ont été développées à cette fin ; on peut les diviser en deux classes : la première consiste à partir d'une situation où tous les interrupteurs sont fermés, et où l'ouverture de chacun d'eux fait l'objet d'un calcul itératif pour en connaître les conséquences sur l'intensité du courant dans les diverses sections du réseau. L'autre classe d'approches part d'un réseau où tous les interrupteurs sont ouverts, et évalue les conséquences de leur fermeture. Plus récemment, des heuristiques basées sur des configurations optimales pour des sous-graphes ont été proposées pour minimiser les pertes Joule ([56] et [39]). Dans [25] par exemple, des permutations sont appliquées sur une configuration initiale afin d'insérer itérativement les fournisseurs dans le réseau.

Dans ce chapitre, on cherche à déterminer une configuration sans avoir recours à la simulation. Pour ce faire, on considère que si la distance entre tout consommateur et son fournisseur d'électricité est inférieure à D_{max} arêtes, alors les pertes Joule et les chutes de tension restent acceptables (voir [25]). La valeur de D_{max} est supposée connue : l'exploitant du réseau de distribution la détermine en fonction de la qualité de service qu'il vise. A ma connaissance, il s'agit de la première tentative de calcul d'une configuration prenant en compte les chutes de tension. Cet aspect du problème semble voué à prendre une importance croissante car les chutes de tension sont un phénomène indésirable qui se manifeste avec d'autant plus d'ampleur que la demande globale d'électricité se rapproche des capacités de production, situation que les experts présentent comme une tendance lourde dans le monde entier. En outre, l'objectif du problème consiste à maximiser la marge de puissance du fournisseur le plus chargé afin de doter le réseau d'une configuration pérenne le protégeant aussi durablement que possible de l'augmentation de la demande [50]. Ce problème est appelé PCRDE pour *Problème de Configuration de Réseau de Distribution d'Electricité*. Certes, il reste possible de changer l'état des interrupteurs en ligne pour faire face à une situation imprévue, mais une telle opération est difficile à mettre en œuvre (voir [29]) et ne paraît envisageable qu'en dernier recours, pour éviter un black out.

La section 3.2 introduit le problème plus formellement et présente des conditions de faisabilité. Une analyse de complexité de PCRDE est conduite et une heuristique gloutonne sans garantie de performance est introduite. Une formulation de PCRDE basée sur les arêtes, dûe à [6] est présentée dans la section 3.3. Une formulation de PCRDE basée sur les sommets est ensuite introduite dans la section 3.4. Les résultats de l'heuristique et ceux des deux formulations précitées résolues avec XPRESS-MP sont ensuite comparés et discutés dans la section 3.5.

3.2 Définition du problème et analyse de complexité

3.2.1 Définition du problème

Le réseau de distribution d'électricité basse tension est constitué de n_f fournisseurs chargés d'alimenter n_c consommateurs. Il est représenté par un graphe non orienté $G(V, E)$ dont l'ensemble V des sommets est partitionné en deux ensembles disjoints : V_f est l'ensemble des fournisseurs ($n_f = |V_f|$) et V_c est l'ensemble des consommateurs ($n_c = |V_c|$). Une arête représente une ligne électrique munie de son interrupteur, et $m = |E|$. Le graphe G est supposé être tel que toute composante connexe contient au moins un fournisseur (dans le cas contraire le problème n'a pas de solution car certains consommateurs ne peuvent pas être alimentés). Chaque sommet de V_f est pondéré par *pow*, qui est la puissance maximale commune à tous les fournisseurs. Tout sommet i de V_c est pondéré par dem_i, qui représente la demande maximale du consommateur i pour tout i dans V_c. La demande maximale est fixée par le type d'abonnement souscrit par le consommateur (par exemple 6 kW). La distance minimale dans G entre deux sommets i et j, notée $d_{i,j}$, est la longueur de la chaîne la plus courte entre i et j exprimée en nombre d'arêtes. Comme G est non orienté, $d_{i,j} = d_{j,i}$ pour tout $(i, j) \in V \times V$. L'ensemble des consommateurs dont la distance minimale au fournisseur j est inférieure ou égale à D_{max} est noté N_j pour tout j dans V_f : N_j est l'ensemble des consommateurs qui peuvent être connectés au fournisseur j. S'il existe un consommateur n'appartenant à aucun N_j, alors PCRDE n'a pas de solution.

Une configuration admissible est un sous-ensemble d'arêtes de E satisfaisant les contraintes suivantes :

1. *Contrainte sur la satisfaction de la demande.* La capacité de chaque fournisseur doit être suffisante pour satisfaire la somme des demandes des consommateurs qui lui sont affectés sous cette configuration,
2. *Contrainte de connexité.* Pour des raisons tenant à la technologie, il ne doit pas exister de chaîne entre deux fournisseurs, ce qui implique que chaque consommateur est connecté à un et un seul fournisseur, et que la configuration est un sous-graphe de G sans circuit.
3. *Contrainte de distance.* Afin de contrôler les pertes Joule et les chutes de tension, la distance maximale entre un consommateur et son fournisseur doit être inférieure ou égale à D_{max}, où la distance est mesurée en nombre d'arêtes.

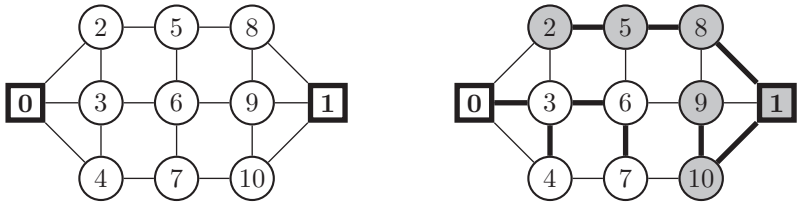
Ainsi, une configuration admissible définit une partition de $V = \cup_{j \in V_f} V_j$ où V_j est l'ensemble des consommateurs alimentés par le fournisseur j, et où la demande que doit satisfaire le fournisseur j est $load_j = \sum_{i \in V_j} dem_i$. Enfin, la quantité M_j est la marge du fournisseur j : $M_j = pow - load_j$. PCRDE consiste à déterminer une configuration admissible maximisant M_{min}, c'est-à-dire la plus petite marge sur l'ensemble des fournisseurs.

3.2.2 Encodage de la solution

Une solution (c'est-à-dire une configuration) de PCRDE pourra être encodée de deux façons. Le premier encodage se fonde sur les arêtes, il est utilisé dans la « formulation par les arêtes » introduite dans la section 3.3. Le second encodage se fonde sur les sommets, il sert de support à la « formulation par les sommets » de PCRDE, introduite dans la section 3.4.

- L'encodage fondé sur les arêtes, qui vise à représenter l'état des interrupteurs est fréquemment utilisé dans la littérature pour aborder les problèmes de conception de réseaux comme on peut le voir dans [1] et [7]. Soit (i, j) une arête de E. Une solution de PCRDE peut être vue comme une forêt de G dans laquelle toute composante connexe contient un et un seul fournisseur. Une solution de PCRDE est donc représentée par la donnée de toutes les arêtes dont l'interrupteur associé est fermé. Tous les autres interrupteurs sont alors supposés être ouverts. Plus formellement, on associe la variable de décision $Q_{i,j}$ à toute arête $(i, j) \in E$, et on pose $Q_{i,j} = 1$ si et seulement si l'interrupteur associé à l'arrête (i, j) est fermé ($Q_{i,j} = 0$ sinon).
- L'encodage fondé sur les sommets représente l'affectation des consommateurs aux fournisseurs. Comme chaque consommateur doit être connecté à un et un seul fournisseur, toute solution à PCRDE peut être vue comme la solution d'un problème d'affectation. Plus formellement, pour tout consommateur $i \in V_c$, et pour tout fournisseur $j \in V_f$, la variable de décision binaire $x_{i,j}$ prend la valeur 1 si et seulement si le consommateur i est alimenté en électricité par le fournisseur j ($x_{i,j} = 0$ sinon).

Deux algorithmes de complexité polynomiale sont proposés pour opérer la transformation d'une solution de PCRDE exprimée dans un encodage vers l'autre. La figure 3.1(a) est une instance de PCRDE. La demande de chaque consommateur est égale à 1, la capacité de chaque fournisseur est égale à 5, et $D_{max} = 3$. La figure 3.1(b) est une solution réalisable ; les interrupteurs fermés apparaissent en trait fort, et les consommateurs alimentés par le fournisseur 1 sont représentés par des sommets sur fond gris. La fonction objectif est ici égale à 0 car la charge du fournisseur 1 est égale à sa capacité, la marge minimale est donc nulle.



(a) Une instance (b) Une solution

FIGURE 3.1 – Une instance et une solution de PCRDE.

La solution proposée peut être écrite comme suit à l'aide de l'encodage fondé sur les arêtes (seules les variables non nulles apparaissent)

$$\begin{array}{lll} Q_{0,3} = 1 & Q_{3,6} = 1 & Q_{3,4} = 1 \\ Q_{6,7} = 1 & Q_{2,5} = 1 & Q_{5,8} = 1 \\ Q_{1,8} = 1 & Q_{1,10} = 1 & Q_{9,10} = 1 \end{array}$$

L'encodage basé sur les sommets de cette solution est

$$\begin{array}{lll} x_{2,1} = 1 & x_{3,0} = 1 & x_{4,0} = 1 \\ x_{5,1} = 1 & x_{6,0} = 1 & x_{7,0} = 1 \\ x_{8,1} = 1 & x_{9,1} = 1 & x_{10,1} = 1 \end{array}$$

Passage d'un encodage de la solution à l'autre

L'algorithme 1 permet de transformer une solution écrite sous l'encodage fondé sur les arêtes en une solution écrite sous l'encodage fondé sur les sommets. L'algorithme 2 opère la transformation réciproque. Il faut noter que l'application successive des deux algorithmes à partir d'une solution écrite sous l'encodage fondé sur les arêtes ne conduit pas nécessairement à la même solution, mais à une solution équivalente dans l'encodage fondé sur les arêtes. Une telle situation est possible car à une solution écrite sous l'encodage fondé sur les sommets peut correspondre différentes solutions écrites sous l'encodage fondé sur les arêtes. En revanche, à une solution écrite sous l'encodage fondé sur les arêtes correspond toujours une unique solution écrite sous l'encodage fondé sur les sommets.

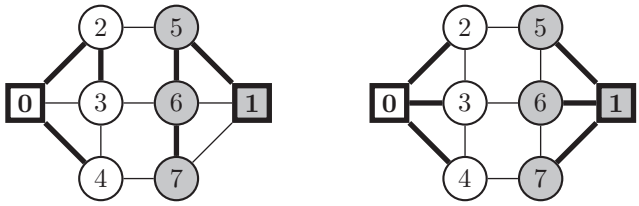
$x_{j,j} \leftarrow 1 \quad \forall j \in V_f$;
$x_{i,j} \leftarrow 0 \quad \forall (i,j) \in V_c \times V_f$;
while $\exists (i,i') \in E | (Q_{i,i'} = 1) \wedge (\sum_{j \in V_f} x_{i,j} = 1) \wedge (\sum_{j \in V_f} x_{i',j} = 0)$ **do**
  $j \leftarrow \sum_{k \in V_f} k x_{i,k}$;
  $x_{i',j} \leftarrow 1$;
end

Algorithme 1: Transformation de l'encodage fondé sur les arêtes vers l'encodage fondé sur les sommets.

$Q_{i,i'} \leftarrow 0 \quad \forall (i,i') \in E$
$L \leftarrow V_f$
while $|L| > 0$ **do**
  $L' \leftarrow \{\emptyset\}$
  forall the $(i,i') \in E | (i \in L) \wedge (i' \notin L') \wedge (Q_{i,i'} = 0) \wedge (\sum_{j \in V_f} jx_{i,j} = \sum_{j \in V_f} jx_{i',j})$ **do**
    $Q_{i,i'} \leftarrow 1$
    $L' \leftarrow L' \cup \{i'\}$
  end
  $L \leftarrow L'$
end

Algorithme 2: Transformation de l'encodage fondé sur les sommets vers l'encodage fondé sur les arêtes.

L'algorithme 2 retourne une solution écrite sous l'encodage fondé sur les arêtes qui minimise la longueur des chaînes entre chaque consommateur et son fournisseur. Il s'agit là d'une caractéristique désirable car la présence de chaînes de longueur non minimale peut être à l'origine de violations de la contrainte de distance. Une telle situation est illustrée ci-après. On considère deux fournisseurs dont la capacité est 3, et six consommateurs dont la demande est égale à 1 avec $D_{max} = 1$. La figure 3.2 montre deux solutions différentes dont les encodages fondés sur les sommets sont identiques. La figure 3.2(a) montre une solution infaisable (la contrainte de distance n'est pas satisfaite pour les consommateurs 3, 6 et 7) alors que la solution de la figure 3.2(b), produite par l'algorithme 2 est admissible.



(a) Une solution non faisable. (b) Une solution faisable.

FIGURE 3.2 – Deux encodages différents fondés sur les arêtes d'une solution associés à un même encodage fondé sur les sommets.

3.2.3 Conditions de faisabilité et analyse de complexité de PCRDE

Cette section établit des conditions nécessaires de faisabilité pour PCRDE, ainsi qu'une condition suffisante dans un cas particulier. Lorsque les conditions nécessaires de faisabilité sont remplies, on montrera que déterminer la faisabilité de PCRDE est un problème de décision $\mathcal{NP}$-complet. Ensuite, on montrera que PCRDE est $\mathcal{NP}$-difficile au sens fort, et une condition nécessaire et suffisante de faisabilité sera fournie dans un cas particulier, ainsi qu'une heuristique pour PCRDE.

Lemme 1. *Chacune des conditions suivantes est une condition nécessaire de faisabilité de* PCRDE.

- *La somme des capacités des fournisseurs est supérieure ou égale à la somme des demandes des consommateurs :*

$$n_f pow \geq \sum_{i \in V_c} dem_i$$

- *La distance minimale de tout consommateur à l'ensemble des fournisseur est inférieure ou égale à D_{max} :*

$$\min_{j \in V_f} d_{i,j} \leq D_{max} \qquad \forall i \in V_c$$

Démonstration. On vérifie immédiatement que si la première condition n'est pas remplie, alors le problème n'a pas de solution car la demande des consommateurs ne peut pas être satisfaite. Si la seconde condition n'est pas remplie, alors c'est la contrainte de distance qui n'est pas satisfaite pour au moins un consommateur et PCRDE n'a pas de solution. □

Lemme 2. *Si les deux conditions nécessaires de faisabilité ci-dessus sont satisfaites, alors la condition suivante est une condition suffisante de faisabilité de* PCRDE *:*

La capacité de chaque fournisseur couvre la somme des demandes de tous les consommateurs appartenant à la même composante connexe que lui dans G.

Démonstration. On suppose que les conditions des lemmes 1 et 2 sont satisfaites. Dans ce cas, une solution faisable pour PCRDE peut être calculée par l'algorithme 3. Cet algorithme retourne une solution sous l'encodage basé sur les sommets. Initialement, chaque fournisseur est affecté à lui-même et aucun consommateur n'est alimenté ($x_{i,j} = 0$ pour tout $(i, j) \in V_c \times V_f$). La charge du fournisseur j, $load_j$, est initialisée à 0 et A_j, l'ensemble des consommateurs qui peuvent être affectés au fournisseur j est initialement vide. L est l'ensemble de tous les sommets dont l'affectation à un fournisseur est connue, et dont la distance à leur fournisseur est égale à $k-1$; au début de la première itération, $L = V_f$. L' est l'ensemble de tous les consommateurs adjacents à L : chaque fois qu'une arête reliant un sommet $i \in L$ à consommateur i' non affecté est identifiée, i' est ajouté à L', l'index j du fournisseur alimentant i est retrouvé, et i' est ajouté à A_j car ce consommateur peut être alimenté par j à travers le sommet i. L est ensuite rendu égal à L' pour préparer l'itération suivante, et les consommateurs de l'ensemble L' sont affectés aux fournisseurs à l'aide de l'algorithme LPT (voir [8, 51]) : le fournisseur dont la charge est minimale est noté j_0, on lui affecte le consommateur i_0 de A_{j_0} ayant la plus forte demande. La charge du fournisseur j_0 est ensuite mise à jour, et i_0 est retiré de L'. Lorsque L' est vide, k est incrémenté de 1 et une nouvelle itération commence à moins que $k > D_{max}$. Ici, k est à la fois un compteur d'itérations, et la distance des sommets de L au fournisseur qui leur est affecté. La capacité des fournisseurs, *pow*, n'intervient pas dans cet algorithme car elle est supposée être suffisamment grande pour couvrir la demande. Lorsque $G(V_f + V_c, E)$ est un graphe biparti complet, PCRDE est identique au problème d'ordonnancement noté $P||C_{max}$ (c'est-à-dire le problème à machines parallèles identiques où l'on cherche à minimiser la date de fin d'exécution de la dernière opération, les opérations n'étant pas interruptibles), et l'algorithme 3 est identique à LPT, qui est une heuristique bien connue pour ce problème d'ordonnancement classique.

Comme on peut le voir, la solution retournée est telle que chaque consommateur est affecté à son plus proche fournisseur. Par conséquent, la solution calculée par l'algorithme 3 est réalisable si et seulement si $\min_{j \in V_f} d_{i,j} \leq D_{max}$, $\forall i \in V_c$. Certains consommateurs ne sont pas alimentés à l'issue de l'algorithme seulement si la première condition de faisabilité du lemme 1 n'est pas satisfaite.

Cet algorithme est polynomial parce que le nombre d'opérations dans la première boucle `while` encapsulée est égal à 3, cette boucle étant répétée au plus $n_c(n_c - 1)$ fois (cette borne est atteinte lorsque $G(V_c, E)$ est complet). Le nombre d'opérations dans la seconde boucle `while` encapsulée est égal à $n_f + n_c + 3$, cette boucle étant répétée moins de n_c fois (car $D_{max} \leq n_c$). Par conséquent, sa complexité est de $\mathcal{O}\left(n_c^3\right)$ opérations. □

Théorème 1. *Si la capacité commune aux fournisseurs est supérieure ou égale à la somme des demandes de tous les consommateurs, alors* PCRDE *est faisable si et seulement si*

$$\min_{j \in V_f} d_{i,j} \leq D_{max} \qquad \forall i \in V_c$$

Démonstration. Premièrement, s'il existe i dans V_c tel que $\min_{j \in V_f} d_{i,j} > D_{max}$, alors PCRDE n'a pas de solution d'après le lemme 1. Deuxièmement, si la capacité commune aux fournisseurs est supérieure ou égale à la somme des demandes de tous les consommateurs, alors la seconde condition de faisabilité du lemme 1 et la condition suffisante de faisabilité du lemme 2 sont remplies. Ainsi, PCRDE est faisable, et dans ce cas l'algorithme 3 peut être utilisé comme une heuristique pour obtenir une solution faisable en temps polynomial. □

Lemme 3. *L'algorithme 3 est sans garantie de performance.*

Démonstration. A la différence de l'algorithme LPT dont la garantie de performance est de $\frac{4}{3}$ (voir [51]), l'algorithme 3 retourne des résultats sans aucune garantie de performance. Dans l'instance de PCRDE présentée à la figure 3.3, la demande des consommateurs 2 et 3 est α et la celle des consommateurs 4 et 5 est β. $D_{max} = 2$ et chaque fournisseur a une capacité de $2(\alpha + \beta)$, par conséquent PCRDE est faisable d'après le lemme 2.

La solution de la figure 3.3(a) est optimale car les deux fournisseurs ont la même marge ($M^*_{min} = \alpha + \beta$). La solution de la figure 3.3(b) a été produite par l'algorithme 3. Cet algorithme, qui construit ses solutions par distance croissante aux fournisseurs n'a pas d'autre choix que d'affecter les consommateurs 2 et 3 au fournisseur 0, et les consommateurs 4 et 5 au fournisseur 1. Par conséquent, la marge du fournisseur 0 est 2β et celle du fournisseur 1

$x_{j,j} \leftarrow 1 \quad \forall j \in V_f$
$x_{i,j} \leftarrow 0 \quad \forall (i,j) \in V_c \times V_f$
$load_j \leftarrow 0 \quad \forall j \in V_f$
$L \leftarrow V_f$
$k \leftarrow 1$
while $\big(k \leq D_{max} \wedge |L| > 0\big)$ **do**
  $L' \leftarrow \{\emptyset\}$
  $A_j \leftarrow \{\emptyset\} \quad \forall j \in V_f$
  `/* Calcul de` L'`, l'ensemble des consommateurs` i' `non affectés à un fournisseur, à la distance` k `du fournisseur le plus proche */`
  while $\exists (i,i') \in E | i \in L \wedge \sum_{k \in V_f} x_{i',k} = 0$ **do**
    $L' \leftarrow L' \cup \{i'\}$
    $j \leftarrow \sum_{k \in V_f} k x_{i,k}$
    $A_j \leftarrow A_j \cup \{i'\}$
  end
  $L \leftarrow L'$
  `/* Allocation des consommateurs de` L' `aux fournisseurs avec l'algorithme LPT */`
  while $|L'| > 0$ **do**
    $j_0 \leftarrow \arg\min_{j | A_j \neq \{\emptyset\}} load_j$
    $i_0 \leftarrow \arg\max_{i \in A_{j_0}} dem_i$
    $x_{i_0,j_0} \leftarrow 1$
    $load_{j_0} \leftarrow load_{j_0} + dem_{i_0}$
    $L' \leftarrow L' \backslash \{i_0\}$
  end
  $k \leftarrow k + 1$
end

Algorithme 3: Une heuristique gloutonne basée sur l'algorithme LPT pour PCRDE lorsque les conditions du lemme 2 sont satisfaites.

(a) Une solution optimale (b) La solution de l'algorithme 3

FIGURE 3.3 – Une instance montrant que l'algorithme 3 est sans garantie de performance.

est 2α. La garantie de performance offerte par l'algorithme 3 dans ce cas s'écrit donc $\frac{\alpha+\beta}{\min(2\alpha,2\beta)}$. Or cette quantité peut être rendue aussi grande que l'on veut.

□

Théorème 2. *Déterminer si* PCRDE *est faisable ou non est un problème $\mathcal{NP}$-difficile au sens fort.*

Démonstration. On prouve le théorème 2 par réduction au problème 3-Partition.

- *Réduction du problème de faisabilité de* PCRDE *à 3-Partition.*
  On rappelle que le problème 3-Partition (voir [30]) est défini par un ensemble A constitué de 3η entiers $\{1,\ldots,3\eta\}$ supérieurs ou égaux à 1, et d'un entier positif B tel que $\frac{B}{4} < a_i < \frac{B}{2}$ pour tout $i \in \{1,\ldots,3\eta\}$ et $\sum_{i=1}^{3\eta} a_i = \eta B$. La question de ce problème de décision est la suivante : *L'ensemble A peut-il être partitionné*

en η sous-ensembles disjoints $A_1, \ldots, A_\eta$ tels que $\sum_{i|a_i \in A_j} = B$ pour tout j in $\{1, \ldots, \eta\}$?
Pour une instance donnée de 3-Partition, une instance particulière de PCRDE est construite comme suit. On considère η fournisseurs dont la capacité est B, et 3η consommateurs dont la demande est a_i pour tout $i \in \{1, \ldots, 3\eta\}$. Le graphe $G(V_f + V_c, E)$ est une graphe biparti complet (c'est-à-dire que tout fournisseur peut être directement connecté à tout consommateur). De plus, $D_{max} = 1$. La question associée est alors : *Existe-t-il une affectation des consommateurs aux fournisseurs telle que la contrainte de capacité de chaque fournisseur est respectée ?*

- *Identification du problème de faisabilité de* PCRDE *à 3-Partition*
  Si 3-Partition a une réponse affirmative, alors il en est de même pour le problème de faisabilité de PCRDE : en effet, une solution faisable pour PCRDE peut être construite comme suit. Le consommateur i est affecté au fournisseur j si et seulement si $a_i \in A_j$. Ainsi, la charge de chaque fournisseur est exactement égale à sa capacité B.
- *Identification de 3-Partition au problème de faisabilité de* PCRDE
  Si le problème de faisabilité de PCRDE a une réponse affirmative, alors il en est de même pour 3-Partition : en effet, une solution de 3-Partition est construite comme suit. Pour tout $j \in \{1, \ldots, \eta\}$, l'ensemble A_j est constitué de tous les entiers a_i tels que le consommateur i est affecté au fournisseur j. Il en résulte que la somme des éléments de A_j est égale à B pour tout $j \in \{1, \ldots, \eta\}$.

□

Théorème 3. PCRDE *est $\mathcal{NP}$-difficile au sens fort.*

Démonstration. Ce résultat est une conséquence directe du théorème 2, car trouver une solution optimale à un problème implique de trouver une solution faisable. Plus spécifiquement, on peut observer que déterminer une solution optimale pour PCRDE est équivalent à trouver la plus petite valeur pour *pow* pour laquelle PCRDE est faisable. Comme déterminer si PCRDE est faisable ou non pour une valeur donnée de *pow* est un problème $\mathcal{NP}$-difficile au sens fort, il en est de même de l'obtention d'une solution optimale de PCRDE. □

La définition du problème peut être modifiée comme suit sans perte de généralité. La capacité des fournisseurs est stockée dans une variable $\overline{pow}$, et *pow* est fixé à la somme des demandes de tous les consommateurs. Ainsi, il suffit que la condition du théorème 1 soit satisfaite pour que PCRDE soit faisable. Le problème peut alors être abordé avec l'algorithme 3 en guise d'heuristique, ou encore avec l'une des approches exactes présentées dans les sections 3.3 et 3.4. Ensuite, la véritable capacité des fournisseurs, $\overline{pow}$, est comparée à $pow - M_{min}$. Si $\overline{pow} \geq pow - M_{min}$, alors PCRDE est faisable et la valeur de son objectif est $\overline{pow} - pow + M_{min}$. Dans le cas contraire, PCRDE n'a pas de solution. L'avantage de cette version modifiée est manifeste lorsque PCRDE n'a pas de solution : dans ce cas, la capacité manquante pour que le problème admette une solution est $pow - M_{min} - \overline{pow}$. Cette information peut être utile à l'exploitant du réseau, qui dans le cas de la version originale du problème ne saurait pas quantifier la mesure dans laquelle le problème est infaisable.

3.3 Formulation par les arêtes de PCRDE

La formulation par les arêtes, notée PCRDE-EF, fait intervenir un graphe orienté $G_d(V, A)$ construit à partir de G comme suit. Chaque arête $(i, j) \in E$ est remplacée par deux arcs (i, j) et (j, i) dans A. Ainsi $|A| = 2|E|$. Comme PCRDE-EF s'appuie sur l'encodage fondé sur les arêtes, calculer une solution revient à déterminer l'état de tous les interrupteurs.

Les variables de décision de PCRDE-EF sont les suivantes :

- M_{min} est la plus petite marge parmi les marges des n_f fournisseurs. La marge d'un fournisseur est la différence entre sa capacité et la somme des demandes des consommateurs qu'il alimente
- Pour tout $(i, j) \in A$, $Q_{i,j}$ est une variables binaire qui est égale à 1 si l'interrupteur (i, j) est fermé et si la puissance circule de i vers j. Si ce n'est pas le cas, alors $Q_{i,j}$ est nulle
- Pour tout $(i, j) \in A$, $f_{i,j}$ est la puissance circulant de i vers j
- Pour tout $i \in V_c$, D_i est la distance séparant le consommateur i du fournisseur qui l'alimente dans la solution courante
- Pour tout $i \in V_f$, D_i est nulle

$$\text{Maximiser} \quad M_{min} \tag{3.1}$$

$$\text{Sous les contraintes} \quad \sum_{i \in V} Q_{i,j} = 1 \qquad \forall j \in V_c \tag{3.2}$$

$$\sum_{i \in V} f_{i,j} - \sum_{k \in V_c} f_{j,k} = dem_j \qquad \forall j \in V_c \tag{3.3}$$

$$f_{i,j} - dem_j \times Q_{i,j} \geq 0 \qquad \forall (i,j) \in V \times V_c \tag{3.4}$$

$$f_{i,j} - \left(\sum_{k \in V_c} dem_k \right) \times Q_{i,j} \leq 0 \qquad \forall (i,j) \in V \times V \tag{3.5}$$

$$pow - \sum_{j \in V_c} f_{i,j} \geq M_{min} \qquad \forall i \in V_f \tag{3.6}$$

$$D_j = 0 \qquad \forall j \in V_f \tag{3.7}$$

$$D_j \geq D_i + 1 - (1 - Q_{i,j}) \times D_{max} \qquad \forall (i,j) \in V \times V_c \tag{3.8}$$

$$D_j \leq D_i + 1 + (1 - Q_{i,j}) \times D_{max} \qquad \forall (i,j) \in V \times V_c \tag{3.9}$$

$$D_i \leq D_{max} \qquad \forall i \in V_c \tag{3.10}$$

$$Q_{i,j} = 0 \qquad \forall (i,j) \notin A \tag{3.11}$$

$$Q_{i,j} \in \{0,1\} \qquad \forall (i,j) \in A \tag{3.12}$$

$$f_{i,j} \geq 0 \qquad \forall (i,j) \in A \tag{3.13}$$

$$M_{min} \geq 0 \tag{3.14}$$

La fonction objectif, (3.1), consiste à maximiser la marge minimale. La contrainte (3.2) assure qu'un et un seul interrupteur est utilisé pour alimenter chaque consommateur. La contrainte (3.3) est une équation de conservation de la puissance en tout consommateur. Les contraintes (3.4) et (3.5) relient les variables $f_{i,j}$ et $Q_{i,j}$ en ne permettant la transmission de puissance entre deux sommets que si $Q_{i,j}$ est égale à 1 et $f_{i,j}$ est au moins égale à dem_j. La contrainte (3.6) définit M_{min} comme la plus petite marge des fournisseurs. Les contraintes (3.7) à (3.10) permettent le calcul de D_i pour tout $i \in V$, et assurent que la contrainte de distance est satisfaite pour tout consommateur. La contrainte (3.11) interdit le transfert de puissance entre deux sommets non adjacents, et les contraintes (3.12) à (3.14) assurent l'intégralité et la non négativité des variables de décision.

PCRDE-EF est inspiré du problème de flot dans les réseaux présenté dans [16], (contraintes (3.3) à (3.5)) mais avec les différences suivantes. En premier lieu, chaque consommateur doit recevoir une quantité de flot en provenance d'un unique fournisseur comme le fait apparaître la contrainte (3.2). En second lieu, en raison de la contrainte de distance, des variables supplémentaires attachées aux sommets (D_i) doivent être introduites (contraintes (3.7) à (3.10)). En troisième lieu, la fonction objectif ne dépend pas du flot circulant dans les arcs du réseau, mais elle vise à maximiser la plus petite marge des fournisseurs (contrainte (3.6)).

La méthode du Simplexe est connue pour être efficace pour traiter certains problèmes de flot dans les réseaux comme le mentionne Chvátal dans [16], et plusieurs versions entières difficiles comme le *single source fixed charge network flow problem* (voir [71]) peuvent être abordées efficacement avec des solvers commerciaux utilisant des stratégies sophistiquées pour générer des coupes, brancher, calculer des bornes, déterminer de bonnes solutions initiales etc. Aussi, PCRDE-EF est-il résolu avec XPRESS-MP ; les résultats numériques obtenus sont présentés dans la section 3.5.

3.4 Formulation par les sommets de PCRDE

La formulation par les sommets, notée PCRDE-VF s'appuie sur l'encodage fondé sur les sommets. Par conséquent, les variables de décision sont les suivantes : pour tout $(i,j) \in V_c \times V_f$, $x_{i,j}$ est fixé à 1 si et seulement si le consommateur i est affecté au fournisseur j, sinon $x_{i,j} = 0$.

$$\text{Maximiser} \quad M_{min} \tag{3.15}$$

$$\text{Sous les contraintes} \quad pow - \sum_{i \in N_j} x_{i,j} dem_i \geq M_{min} \qquad \forall j \in V_f \tag{3.16}$$

$$\sum_{j \in V_f} x_{i,j} = 1 \qquad \forall i \in V_c \tag{3.17}$$

$$x_{i,j} = 0 \qquad \forall j \in V_f, \forall i \notin N_j \tag{3.18}$$

$$\text{Les consommateurs affectés au fournisseur } j \text{ forment une composante connexe} \qquad \forall j \in V_f \tag{3.19}$$

$$\text{La distance du consommateur } i \text{ à son fournisseur est inférieure à } D_{max} \qquad \forall i \in V_c \tag{3.20}$$

$$x_{i,j} \in \{0,1\} \qquad \forall (i,j) \in V_c \times V_f \tag{3.21}$$

$$M_{min} \geq 0 \tag{3.22}$$

La fonction objectif, (3.15), est la même que pour PCRDE-EF. La contrainte (3.16) est analogue à la contrainte (3.6) dans PCRDE-EF, elle définit M_{min} comme la marge minimale des fournisseurs. La contrainte (3.17) assure que tout consommateur est affecté à un et un seul fournisseur. La contrainte (3.18) interdit qu'un consommateur ne soit affecté à un fournisseur situé à une distance strictement supérieure à D_{max}. Les contraintes (3.19) et (3.20) sont difficiles à mettre en œuvre, elles sont discutées en détail dans le reste de cette section. Enfin, les contraintes (3.21) et (3.22) assurent l'intégralité et non négativité des variables de décision.

Les contraintes de connexité peuvent être exprimées comme suit. Soit S_j un sous-ensemble de consommateurs de N_j, tel que leur distance à j soit supérieure ou égale à 2. Les consommateurs de S_j peuvent être alimentés par le fournisseur j, mais ne sont pas directement adjacent à j. Soit S un sous-ensemble de S_j. On note CS l'ensemble des consommateurs de N_j voisins de S mais qui ne sont pas dans S. Ainsi, la connexité de la solution est assurée par la contrainte :

$$x_{i,j} \leq \sum_{k \in CS} x_{k,j} \qquad \forall i \in S, \forall S \subseteq S_j, \forall j \in V_f$$

Malheureusement, le nombre de ces contraintes croît de façon exponentielle avec la taille de l'instance, car le nombre d'ensembles S à considérer est égal à la factorielle de $|S_j|$. On se heurte à la même difficulté lorsqu'on aborde le problème du voyageur de commerce par programmation linéaire en variables entières comme le montre [48]. Cependant, appliquer un algorithme de plans coupants classique n'est pas suffisant ici car les contraintes de distance (3.20) doivent également être satisfaites. Un algorithme de plans coupants adapté à PCRDE est proposé dans la section 3.4.1.

3.4.1 Approche de résolution

L'approche proposée pour résoudre PCRDE-VF peut être résumée comme suit. Le programme linéaire en variables mixtes défini par les équations (3.15) à (3.18), (3.21) et (3.22) est renforcé par ce que j'appelle ici des *inégalités de couche* (voir section 3.4.2) puis résolu. Si les contraintes de connexité et de distance se trouvent être satisfaites (c'est-à-dire si les contraintes (3.19) et (3.20) sont vérifiées), alors une solution optimale de PCRDE-VF a été trouvée. Sinon, les coupes correspondant aux contraintes de connexité et de distance violées dans la solution courante (voir section) sont ajoutées à la formulation du problème et le processus de résolution se répète. Cette approche a été employée avec succès dans [46] pour aborder le *minimum power symmetry connectivity problem*, et and [44] pour traiter le *minimum power multicasting problem* dans le cadre de l'optimisation des réseaux sans fil. Ces deux problèmes et PCRDE ont en commun les caractéristiques suivantes : la fonction objectif ne fait référence qu'à des quantités attachées aux sommets, alors que des contraintes difficiles (c'est-à-dire relatives à la connexité) doivent être vérifiées par les arêtes. Ici, un algorithme de plans coupants généralise l'approche itérative proposée par [46] et [44] en assurant conjointement la satisfaction des contraintes de connexité et de distance. De plus, les inégalités de couche sont introduites pour accélérer la convergence de l'algorithme proposé (voir section 3.4.2).

Les grandes étapes de l'algorithme de plans coupants proposé pour aborder PCRDE-VF sont illustrées par le logigramme de la figure 3.4. Initiallement, le problème maître est constitué des équations (3.15) à (3.18), (3.21), (3.22) et aussi (3.23), qui sont les inégalités de couche. Ces contraintes, bien que généralement insuffisantes à elles

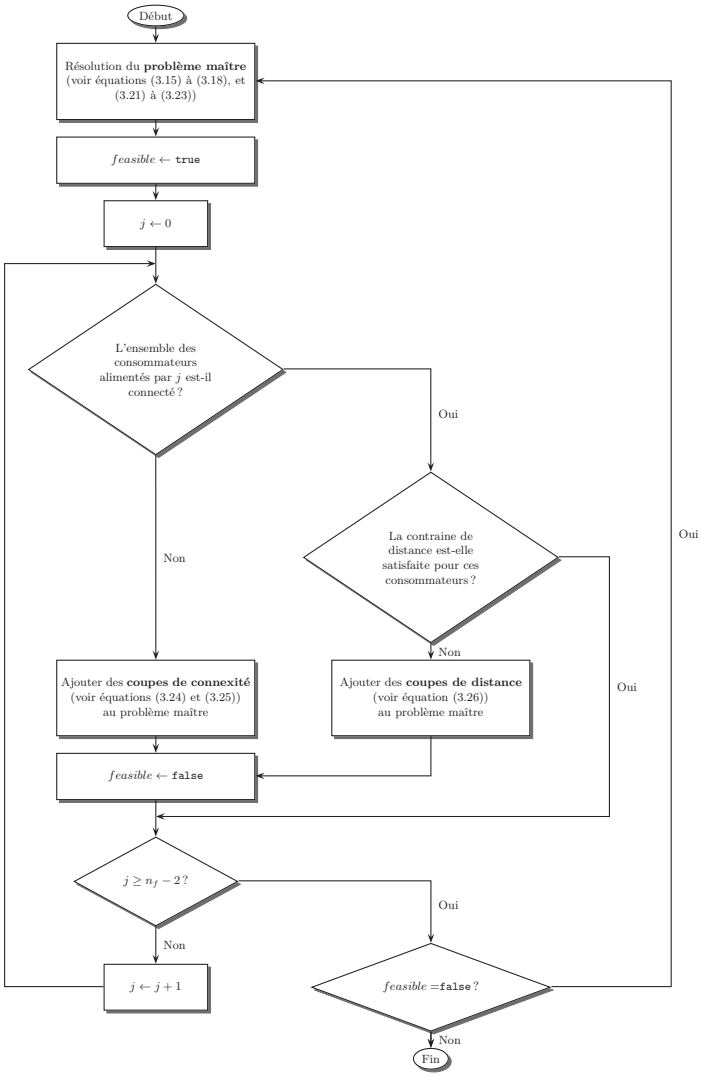


FIGURE 3.4 – Logigramme de l'algorithme de plans coupants pour résoudre PCRDE-VF.

seules pour assurer le respect des contraintes de connexité et de distance, peuvent être vues comme des conditions nécessaires à leur satisfaction, permettant d'accélérer l'approche proposée.

A chaque itération, le problème maître est résolu, la variable binaire $feasible$ prend la valeur `true` et on vérifie si les consommateurs alimentés par le même fournisseur forment une composante connexe. Si ce n'est pas le cas, des coupes de connexité (voir section 3.4.2) sont ajoutées au problème et $feasible$ se voit affecter la valeur `false`. Si les consommateurs alimentés par le fournisseur courant forment une composante connexe, alors on vérifie que la distance séparant chaque consommateur au fournisseur est inférieure ou égale à D_{max}. Si la contrainte de distance n'est pas satisfaite, des coupes de distance (voir section 3.4.2) sont ajoutées au problème et $feasible$ se voit affecter la valeur `false`. Sinon, aucune coupe n'est ajoutée pour le fournisseur courant. Une nouvelle itération commence si $feasible =$ `false`, c'est-à-dire aussi longtemps que toutes les contraintes de connexité et de distance ne sont pas satisfaites par la solution courante. Lorsque l'algorithme s'arrête, la solution optimale au problème maître est également optimale pour PCRDE-VF car le problème maître enrichi des coupes de connexité et de distance est une relaxation de PCRDE-VF.

3.4.2 Prise en compte des contraintes de connexité et de distance

Inégalités de couche

Un algorithme de plans coupants est d'autant plus efficace que la formulation du problème maître est forte. L'introduction d'inégalités de couche vise à diminuer le nombre d'itérations nécessaires à la convergence de l'algorithme de plans coupants en interdisant l'apparition de solutions « trivialement » non faisables pour PCRDE-VF. A cette fin, un ensemble d'inégalités valides (voir [71]) est ajouté pour renforcer le problème maître.

On dit que les consommateurs dont la distance au fournisseur j est égale à λ appartiennent à la couche $L_{j,\lambda}$. Ainsi, pour tout $j \in V_f$ pour tout $\lambda \in \{1, \ldots, D_{max} - 1\}$, tout consommateur $i \in L_{j,\lambda+1}$ peut être affecté au fournisseur j si au moins un consommateur $k \in L_{j,\lambda}$ est également alimenté par le fournisseur j. Cette contrainte peut être écrite comme suit.

$$x_{i,j} \leq \sum_{k \in L_{j,\lambda}} x_{k,j} \qquad \forall i \in L_{j,\lambda+1}, \forall \lambda \in \{1, \ldots, D_{max} - 1\}, \forall j \in V_f$$

Ces inégalités assurent le respect de conditions nécessaires à la connexité en prenant partiellement en compte les contraintes de distance. Elles peuvent facilement être renforcées en observant que pour tout i dans $L_{j,\lambda+1}$, seuls les consommateurs $k \in L_{j,\lambda}$ satisfaisant la condition suivante jouent un rôle dans le respect de la connexité : il existe une chaîne de k à i dont la longueur est inférieure ou égale à $D_{max} - \lambda$, et tels que les consommateurs qui constituent cette chaîne sont tous situés à une distance de j qui est strictement supérieure à λ (sauf pour k). En effet, s'il n'existe pas de chaîne de k à i de longueur inférieure ou égale à $D_{max} - \lambda$, alors le consommateur k n'est pas utile à l'établissement d'une chaîne de i à j. De plus, si toutes les chaînes de k à i contiennent a sommet $v \neq i$ dont la distance est inférieure ou égale à λ, alors le consommateur i peut être alimenté par le fournisseur j par l'intermédiaire du sommet v, ce qui implique que le consommateur k n'est d'aucune aide pour assurer la connexité de i à j.

Les deux ensembles suivants, $R_{k,j}$ et $P_{i,j}$, sont nécessaires à l'écriture des inégalités de couche. Soit k un consommateur appartenant à $L_{j,\lambda}$. L'ensemble des consommateurs dont la distance au fournisseur k est strictement supérieure à λ, et dont la distance à k est inférieure ou égale à $D_{max} - \lambda$ est appelé $R_{k,j}$. Cet ensemble peut être vu comme l'ensemble des consommateurs qui peuvent être alimentés par le fournisseur j par l'intermédiaire de k tout en satisfaisant la contrainte de distance. Soit i un consommateur appartenant à la couche $L_{j,\lambda+1}$, l'ensemble de ses prédécesseurs potentiels dans $L_{j,\lambda}$ est noté $P_{i,j}$. Ainsi, au moins un consommateur de $P_{i,j}$ doit être alimenté par le fournisseur j si i est aussi alimenté par j.

Plus formellement,

$$\begin{aligned} R_{k,j} &= \{v | \lambda < d_{v,j} \wedge d_{v,k} \leq D_{max} - \lambda\} \qquad \forall k \in L_{j,\lambda}, \forall \lambda \in \{1, \ldots, D_{max} - 1\}, \forall j \in V_f \\ P_{i,j} &= \{k \in L_{j,\lambda} | i \in R_{k,j}\} \qquad \forall i \in L_{j,\lambda+1}, \forall \lambda \in \{1, \ldots, D_{max} - 1\}, \forall j \in V_f \end{aligned}$$

Les inégalités de couche sont définies par

$$x_{i,j} \leq \sum_{k \in P_{i,j}} x_{k,j} \qquad \forall i \in L_{j,\lambda+1}, \forall \lambda \in \{1, \ldots, D_{max} - 1\}, \forall j \in V_f \tag{3.23}$$

Comme tout consommateur apparaît dans au plus une couche $L_{j,\lambda}$ pour tout fournisseur $j \in V_f$, le nombre d'inégalités de couche est au plus $n_c \times n_f$.

Coupes de connexité

Les inégalités de couche sont utiles mais généralement insuffisantes pour garantir la connexité. C'est la raison pour laquelle les coupes de connexité sont ajoutées au problème maître chaque fois que la solution courante ne satisfait pas les contraintes de connexité.

Chaque fois que le problème maître est résolu, CC_j, la composante connexe associée au fournisseur j est calculée pour tout $j \in V_f$. L'ensemble des sommets qui sont affectés au fournisseur j mais qui ne sont pas dans CC_j est noté O_j.

Si O_j est non vide, alors les consommateurs affectés au fournisseur j ne forment pas une composante connexe, et une coupe de connexité est requise car la solution courante du problème maître n'est pas faisable pour PCRDE.

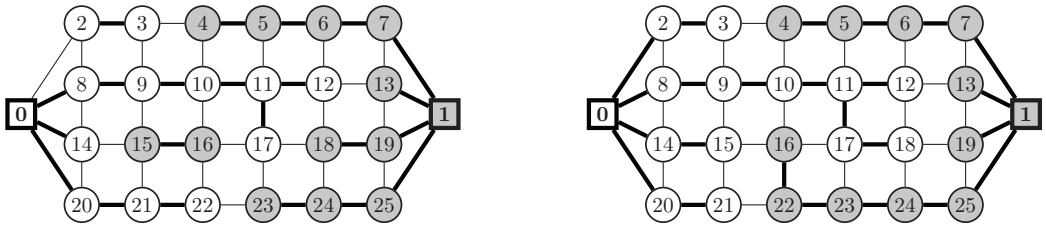
Soit CS_j l'ensemble des consommateurs adjacents à CC_j mais qui sont affectés à un fournisseur différent de j. CS_j peut être vu comme un ensemble d'articulation : retirer ces sommets de G conduirait à isoler les sommets de CC_j du reste du graphe, et en particulier de O_j. Par conséquent, pour que les consommateurs appartenant à O_j puissent être alimentés par j, au moins un consommateur de CS_j doit également être affecté au fournisseur j. Cela peut être écrit comme suit.

$$\sum_{i \in O_j} x_{i,j} \leq |O_j| \sum_{i \in CS_j} x_{i,j} \qquad \forall j \in V_f \tag{3.24}$$

L'inégalité (3.24) est appelée coupe de connexité. En théorie, une solution faisable pour PCRDE peut être atteinte en ajoutant une seule coupe de connexité pour chaque fournisseur j tel que O_j est non vide. Cependant, procéder ainsi ne serait pas aussi efficace que de générer autant de coupes qu'il existe de composantes connexes dans O_j. Il est ainsi nécessaire de construire $\mathcal{C}_j$, l'ensemble de toutes les composantes connexes de O_j. Soit CC une composante connexe de $\mathcal{C}_j$, et CS l'ensemble des consommateurs voisins de CC mais qui ne sont pas affectés à j. Alors, pour qu'un consommateur quelconque de CC puisse être effectivement alimenté par j, au moins un consommateur de CS doit lui aussi être affecté au fournisseur j. Cette coupe de connexité simplifiée s'écrit :

$$x_{i,j} \leq \sum_{k \in CS} x_{k,j} \qquad \forall i \in CC, \forall CC \in \mathcal{C}_j, \forall j \in V_f$$

La figure 3.5(a) illustre ces coupes et montre qu'elles peuvent facilement être renforcées. Le fournisseur 1 et les consommateurs qu'il est censé alimenter ne forment pas une composante connexe car les consommateurs 15 et 16 sont isolés.



(a) Une coupe de connexité est requise.

(b) Une coupe de distance est requise.

FIGURE 3.5 – Illustration de la nécessité des coupes de connexité et de distance.

Les coupes de connexité simplifiées sont :

$$\begin{aligned} x_{15,1} &\leq x_{17,1} + x_{10,1} + x_{22,1} + x_{9,1} + x_{21,1} + x_{14,1} \\ x_{16,1} &\leq x_{17,1} + x_{10,1} + x_{22,1} + x_{9,1} + x_{21,1} + x_{14,1} \end{aligned}$$

En effet, $O_1 = \{15, 16\}$ et $\mathcal{C}_1 = \{O_1\}$, d'où $CC = O_1$ avec l'ensemble d'articulation $CS = \{17, 10, 22, 9, 21, 14\}$. Cependant, pour $D_{max} = 5$ on peut observer que le consommateur 14 ne peut pas être alimenté par le fournisseur 1 car sa distance au sommet 1 est de 6. Par le même raisonnement, on déduit que les consommateurs 9 et 21, dont la distance au fournisseur 1 est égale à 5, ne peuvent être utilisés pour connecter les consommateurs 15 et 16 : même si 9 et 21 étaient alimentés par le fournisseur 1, la longueur de la chaîne la plus courte de 1 à 15 et de 1 à 16 via 9 ou 21 dépasserait D_{max}. Par conséquent, seuls les consommateurs 17, 10 et 22 peuvent aider à connecter 16 à CC. Et enfin, seul le consommateur 17 peut aider à connecter le consommateur 15, car la distance minimale (dans G) entre le fournisseur 1 et les consommateurs 10 et 22 est égale à 4 alors que la distance minimale (dans G) de 10 à 15 et de 22 à 15 est égale à 2. Comme $4 + 2 > D_{max}$, il n'existe pas de chaîne du fournisseur 1 au consommateur 15 via 10 ou 22 respectant la contrainte de distance. On en déduit que :

$$\begin{aligned} x_{15,1} &\leq x_{17,1} \\ x_{16,1} &\leq x_{17,1} + x_{10,1} + x_{22,1} \end{aligned}$$

Les coupes de connexité sont construites comme suit. Afin de ne pas générer un nombre de coupes excessivement grand, une seule coupe de connexité est ajoutée au problème maître pour chaque composante connexe CC de O_j, pour tout $j \in V_f$. Un consommateur $i_{CC} \in CC$ dont la distance à j est minimale est choisi arbitrairement. Ensuite, tous les consommateurs k de CC tels que $d_{k,j} + d_{i_{CC},k} > D_{max}$ sont retirés de CS. On rappelle que $d_{k,j}$ est la distance minimale du fournisseur j au consommateur k dans G (cette distance n'est pas mesurée sur le graphe induit par la solution courante), et $d_{i_{CC},k}$ est la distance minimale de i_{CC} à k dans G. Comme on l'a vu, de tels consommateurs ne peuvent servir d'intermédiaire pour connecter i_{CC} à CC_j au cours de la prochaine itération de l'algorithme de plans coupants. Plus formellement, la coupe de connexité associée à CC est :

$$x_{i_{CC},j} \leq \sum_{\substack{k \in CS \\ d_{k,j}+d_{i_{CC},k} \leq D_{max}}} x_{k,j} \qquad \forall CC \in \mathcal{C}_j, \forall j \in V_f \tag{3.25}$$

Coupes de distance

Pour tout fournisseur $j \in V_f$, les contraintes de distance ne sont vérifiées que lorsque j et les consommateurs qu'il alimente forment une composante connexe. A première vue, il peut paraître surprenant que la contrainte de distance puisse être violée si tous les consommateurs affectés au fournisseur j sont choisis dans N_j (c'est-à-dire parmi ceux dont la distance à j est inférieure ou égale à D_{max} dans G). Cependant, une telle situation se produit lorsque certains consommateurs sur la chaîne la plus courte d'un fournisseur à un consommateur lointain empêchent d'utiliser cette chaîne la plus courte (en raison de leur affectation à un fournisseur différent), ce qui a pour effet d'augmenter la distance de la chaîne la plus courte entre un fournisseur et un consommateur qu'il alimente. La figure 3.5(b) illustre un tel cas avec le fournisseur 0 et le consommateur 18 (où $D_{max} = 5$).

L'unique chaîne de longueur minimale du fournisseur 0 au consommateur 18 a pour longueur 5 dans G. Cependant, comme le consommateur 16 est alimenté par le fournisseur 1, la chaîne de longueur minimale dans G ne peut pas être utilisée ce qui place le consommateur à la distance $6 > D_{max}$ dans la solution du problème maître comme l'illustre la figure 3.5(b).

Comme les contraintes de distance pour le fournisseur j ne sont vérifiées que dans le cas où les contraintes de connexité sont satisfaites pour ce même fournisseur, O_j est vide mais l'ensemble d'articulation CS_j est néanmoins calculé. Soit F_j l'ensemble des consommateurs affectés à j, mais dont la distance à j est égale à $D_{max} + 1$.

Une version simplifiée des coupes de distance impose que pour permettre à tout consommateur i appartenant à F_j de se trouver à une distance de j inférieure ou égale à D_{max} lors de la prochaine itération, au moins un consommateur dans CS_j doit être alimenté par j :

$$x_{i,j} \leq \sum_{k \in CS_j} x_{k,j} \qquad \forall i \in F_j, \forall j \in V_f$$

Dans l'exemple de la figure 3.5(b), cela conduit à

$$x_{18,0} \leq x_{4,0} + x_{16,0} + x_{22,0} + x_{5,0} + x_{23,0} + x_{6,0} + x_{24,0} + x_{13,0} + x_{19,0}$$

Cette coupe est renforcée de la même manière que pour les coupes de connexité : les consommateurs 13 et 19 sont écartés car ils n'appartiennent pas à N_0. Le fait que les consommateurs 6 et 24 soient alimentés par le fournisseur 0, n'est d'aucun secours pour les contraintes de distance car ces deux sommets sont eux-mêmes à une distance de 0 égale à D_{max}. La distance minimale de 0 à 5 et 23 est de 4, mais la distance minimale de ces consommateurs au sommet 18 est de 3 et 2 respectivement, ce qui est trop pour satisfaire la contrainte de distance du consommateur 18 par l'intermédiaire de 5 et 23. Les sommets 4 et 22 peuvent être écartés pour la même raison, ce qui conduit à une coupe beaucoup plus forte : $x_{18,0} \leq x_{16,0}$.

Les coupes de distance sont définies comme suit :

$$x_{i,j} \leq \sum_{\substack{k \in CS_j \\ d_{k,j}+d_{i,k} \leq D_{max}}} x_{k,j} \qquad \forall i \in F_j, \forall j \in V_f \tag{3.26}$$

3.5 Résultats numériques

3.5.1 Génération des instances utilisées

Les instances utilisées pour comparer les différents algorithmes proposés ici sont divisées en trois classes :

- *Instances d'Aoki.* Dix instances sont générées à partir du réseau dont la topologie est présentée dans [42]. Une demande aléatoire entière est générée dans l'ensemble $\{1, \ldots, 100\}$ pour chaque consommateur, et la capacité de chaque fournisseur est égale à la somme des demandes des consommateurs. Cela permet d'assurer que PCRDE est faisable pour toutes ces instances.
- *Instances mimétiques.* Quatre-vingt-dix instances sont générées comme suit. Les fournisseurs sont répartis à intervalles réguliers sur un cercle centré au milieu d'un carré de dimension 100×100, alors que les coordonnées des consommateurs sont générées de manière aléatoire dans le même carré, en générant des segments et des carrés interconnectés. L'objectif est de produire des graphes dont la topologie imite celle des instances d'Aoki (en termes de degré moyen des sommets), comme on peut le voir dans [25] et [42]. Comme pour les instances d'Aoki, la demande des consommateurs est générée aléatoirement, et chaque fournisseur a une capacité égale à la somme des demandes des consommateurs.
- *Instances carrées.* Quatre-vingt-dix instances sont générées comme suit. Si l'on excepte la topologie, la méthode de génération est identique à celle des instances mimétiques. Les graphes ont une structure en forme de maille carrée : les $n_c = n \times n$ consommateurs sont espacés régulièrement dans un carré de côté 100, et chaque consommateur est connecté à tous ses voisins (consommateurs et fournisseurs) situés à une distance inférieure ou égale à $\frac{100}{n+1}$.

La figure 3.6 donne un aperçu des trois classes d'instances utilisées.

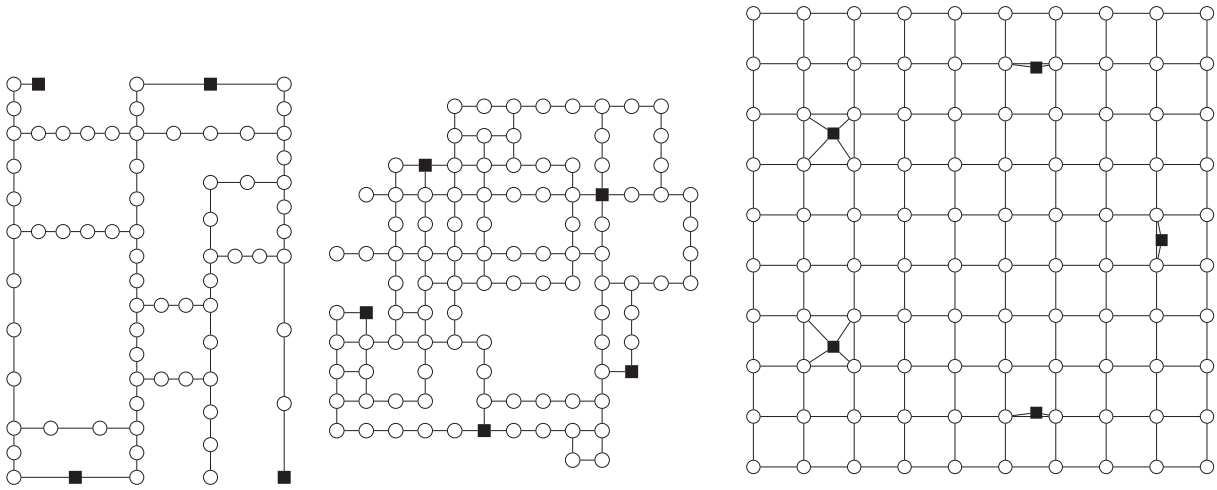


FIGURE 3.6 – Aperçu des instances d'Aoki, des instances mimétiques, et des instances carrées.

3.5.2 Résultats

La formulation par les arêtes, la formulation par les sommets et l'algorithme 3 (qui est utilisé comme une heuristique) sont comparés sur ces trois classes d'instances. L'ordinateur employé utilise un processeur Intel Pentium IV cadencé à 3.2 GHz muni de 1 gigaoctet de RAM sous Microsoft Windows XP, le solver est XPRESS-MP utilisé au travers de la librairie XPRESS-BCL [72].

Chaque ligne dans les tables 3.1, 3.2 et 3.3 correspond à 10 instances de même taille. Les trois premières colonnes donnent le nombre de fournisseurs, de consommateurs et le degré moyen des graphes calculé sur les 10 instances. La colonne *Opt.* indique le nombre d'instances pour lesquelles la solution optimale a été trouvée avec PCRDE-EF (colonne 4) et avec PCRDE-VF (colonne 6) dans la limite d'une heure de calcul par instance. La colonne *CPU* est le temps d'exécution moyen de l'algorithme correspondant exprimé en secondes, calculé seulement pour les instances dont la solution optimale a été trouvée en moins d'une heure. La colonne *Iter.* est le nombre moyen d'itérations de l'algorithme de plans coupants, calculé seulement pour les instances dont la solution optimale a été trouvée. La dernière colonne, *Dev.*, donne la dégradation de performance moyenne de la solution retournée par l'algorithme 3.

Lorsque la solution optimale n'est pas disponible, on utilise la valeur de la fonction objectif retournée à la dernière itération de PCRDE-VF, car cela constitue une borne supérieure de la valeur de l'objectif de PCRDE. La table 3.1 indique par exemple que la qualité des solutions retournées par l'algorithme 3 est en moyenne 14.63% plus mauvaise (c'est-à-dire plus petite) que la valeur optimale de la fonction objectif sur les 10 instances d'Aoki. Comme le temps de calcul requis par l'algorithme 3 est toujours inférieur à une seconde, il n'est pas reporté dans les tables.

TABLE 3.1 – Résultats pour les 10 instances d'Aoki.

Caractéristiques des instances			PCRDE-EF		PCRDE-VF			Heuristique
n_f	n_c	Deg.	Opt.	CPU	Opt.	CPU	Iter.	Dev.
4	68	2.17	10	145.44	10	0.42	0.20	14.63%

TABLE 3.2 – Résultats pour les 90 instances mimétiques.

Caractéristiques des instances			PCRDE-EF		PCRDE-VF			Heuristique
n_f	n_c	Deg.	Opt.	CPU	Opt.	CPU	Iter.	Dev.
3	25	2.41	10	3.49	10	0.21	1.60	15.83%
4	25	2.45	10	1.05	10	0.35	1.70	8.00%
5	25	2.39	10	0.44	10	0.28	1.30	10.87%
3	50	2.58	6	528.18	10	0.08	1.50	14.60%
4	50	2.49	8	350.27	10	2.30	2.70	14.09%
5	50	2.46	9	33.93	10	8.86	2.70	11.67%
3	100	3.07	0	–	10	0.29	1.30	3.84%
4	100	2.84	0	–	10	0.29	2.30	9.64%
5	100	2.62	0	–	10	2.27	4.40	9.40%

TABLE 3.3 – Résultats pour les 90 instances carrées.

Caractéristiques des instances			PCRDE-EF		PCRDE-VF			Heuristique
n_f	n_c	Deg.	Opt.	CPU	Opt.	CPU	Iter.	Dev.
3	25	3.43	10	1389.84	10	76.77	23.00	4.07%
4	25	3.31	10	315.67	10	73.08	8.00	2.84%
5	25	3.73	10	663.41	10	380.63	10.30	6.35%
3	49	3.54	2	252.99	10	21.24	45.90	3.50%
4	49	3.47	1	503.59	10	180.82	40.20	1.65%
5	49	3.56	0	–	7	1395.10	36.43	3.02%
3	100	3.69	0	–	10	3.41	18.80	3.08%
4	100	3.62	0	–	10	24.32	39.80	1.04%
5	100	3.70	0	–	8	1129.48	127.13	3.70%

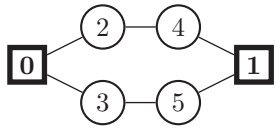
3.5.3 Discussion

PCRDE-EF et PCRDE-VF permettent de déterminer une solution optimale pour toutes les instances d'Aoki comme le montre la table 3.1. Le temps de calcul requis par PCRDE-VF est substantiellement moins grand que pour PCRDE-EF sur cette classe d'instances, et le très faible nombre d'itérations est la conséquence de l'emploi des inégalités de couche qui permettent d'écarter nombre de solutions non faisables. La qualité des solutions retournées par l'algorithme 3 est assez mauvaise, en raison sans doute de la topologie du graphe : le degré moyen des sommets étant bas (2.17), l'algorithme 3 a peu d'options pour tenter d'équilibrer la charge des fournisseurs.

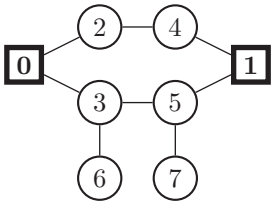
Les résultats sur les instances mimétiques (table 3.2) confirment l'analyse formulée à partir des instances d'Aoki, et permet de l'affiner. Là encore, PCRDE-VF se révèle plus efficace que PCRDE-EF grâce aux inégalités de couche permettant à l'algorithme de converger rapidement. On peut cependant observer que le temps de calcul de PCRDE-VF augmente avec le nombre de fournisseurs (car le nombre de variables de décision augmente lui aussi), alors que le temps de calcul de PCRDE-EF diminue. Comme on pouvait s'y attendre, l'efficacité de l'algorithme 3 semble s'améliorer avec l'augmentation du degré moyen du graphe d'une part, et avec la taille du graphe d'autre part.

Les instances carrées ont la particularité d'avoir un degré moyen sensiblement plus élevé que les instances des deux classes précédentes. A taille de graphe égale, la comparaison entre les tables 3.2 et 3.3 montre l'impact de ce paramètre sur la difficulté du problème pour les deux approches exactes. PCRDE-VF est encore une fois beaucoup plus efficace que PCRDE-EF en termes de nombre de solutions optimales trouvées comme en termes de temps de calcul. Il peut paraître surprenant de lire dans la table 3.3 que PCRDE-VF est plus rapide pour 100 consommateurs que pour 49. Cela illustre le rôle de D_{max} (qui vaut 6 dans les deux cas) sur l'algorithme de plans coupants. En effet, la part des consommateurs qui peuvent être alimentés par au moins deux fournisseurs est plus élevée pour $n_c = 49$ que pour $n_c = 100$: si le consommateur i ne peut être affecté qu'au fournisseur j, alors cela implique qu'un certain nombre d'autres consommateurs situés entre i et j doivent aussi être alimentés par le fournisseur j. Ainsi, ces consommateurs « distants et captifs » induisent-ils implicitement des contraintes qui réduisent l'espace des solutions.

Cette situation est illustrée par la figure 3.7, où $D_{max} = 2$ pour les deux instances proposées. Dans la figure 3.7(a), 4 consommateurs ont une demande de 1 (les fournisseurs ont donc une capacité de 4). Le degré moyen du graphe de la figure 3.7(a) est égal à deux (c'est un graphe cyclique), et le problème maître admet 3 solutions optimales différentes (toutes sont faisables pour PCRDE), qui sont données dans la table 3.4. Dans la figure 3.7(b), deux consommateurs supplémentaires sont ajoutés, sans que le degré moyen du graphe ne soit affecté, et la capacité des fournisseurs est maintenant de 6. Dans ce cas, le problème maître n'admet qu'une seule solution optimale. En effet, en raison de la contrainte de distance, le consommateur 6 ne peut être affecté qu'au fournisseur 0, ce qui implique que le consommateur 3 soit lui aussi alimenté par ce fournisseur. De même, le consommateur 5 ne peut être affecté qu'au fournisseur 1, à cause de la contrainte de distance appliquée au consommateur 7. Enfin, les consommateurs 2 et 4 doivent être affectés aux fournisseurs 0 et 1 respectivement pour que la solution soit optimale. Alors que le temps de calcul de PCRDE-EF tend à diminuer lorsque le nombre de fournisseur augmente, cet effet bénéfique est lourdement contrebalancé par l'effet de l'augmentation de la taille des instances comme le montre la table 3.3 pour $n_f = 49$. Pour ce qui concerne l'algorithme 3, on observe à nouveau que la qualité des solutions qu'il retourne tend à s'améliorer lorsque la taille des instances augmente.



(a) Instance à trois solutions pour $D_{max} = 2$.



(b) Instance admettant une solution unique pour $D_{max} = 2$.

FIGURE 3.7 – Deux instances illustrant le rôle de D_{max} sur l'espace des solutions.

TABLE 3.4 – Trois solutions pour l'instance de la figure 3.7(a).

Solution 1	$x_{0,2} = x_{0,3} = x_{1,4} = x_{1,5} = 1$
Solution 2	$x_{0,2} = x_{0,4} = x_{1,5} = x_{1,3} = 1$
Solution 3	$x_{0,3} = x_{0,5} = x_{1,4} = x_{1,2} = 1$

En définitive, la structure du graphe apparaît comme le second paramètre prépondérant (après le nombre de consommateurs) ayant un impact sur la difficulté de résolution d'une instance. Or en général, on observe que les réseaux de distribution d'électricité sont conçus de façon telle que l'affectation des consommateurs aux fournisseurs est un problème assez fortement contraint. Seules quelques arrêtes additionnelles offrent un peu de flexibilité pour faire face aux black outs et permettre les opérations de maintenance sur le réseau [25]. Pour des raisons évidentes

de coût, ces arêtes additionnelles ne sont pas très nombreuses, de sorte que le nombre de cycles est relativement bas (comme dans les instances d'Aoki). Les tests numériques suggèrent que les approches exactes proposées se comportent bien sur les graphes creux, en particulier PCRDE-VF. De plus, ces instances semblent être celles pour lesquelles l'algorithme 3 retourne des solutions de qualité moyenne, ce qui fait apparaître PCRDE-VF comme la formulation la plus adaptée à la résolution de PCRDE sur des instances réelles.

3.6 Conclusion

Deux approches exactes et une heuristique sont proposées pour aborder le problème de la configuration des réseaux de distribution d'électricité basse tension. La formulation par les sommets semble être très efficace lorsque la topologie du réseau est un graphe relativement creux, ce qui est souvent le cas en pratique.

Les coupes de connexité générées dans le cadre de la résolution de PCRDE sont actuellement focalisées sur un seul consommateur dans chaque composante connexe de la solution du problème maître. Il pourrait être intéressant de choisir les consommateurs sur la base du nombre de fois où ils ont fait partie de composantes connexes ne contenant pas de fournisseur lors des itérations précédentes. Ainsi, on pourrait espérer qu'une focalisation des coupes sur ces consommateurs puisse permettre de décider assez tôt de leur affectation.

Les résultats numériques montrent que l'algorithme de plans coupants voit ses performances se dégrader avec la taille et la densité des instances, alors que la qualité des solutions de l'heuristique suit la tendance inverse. De plus, si l'algorithme de plans coupants n'a pas trouvé de solution optimale à l'issue du temps imparti, il ne retourne aucune solution au problème. Par conséquent, la proposition d'une matheuristiques combinant les avantages des deux approches apparaît prometteuse. A cette fin, il faudrait alors écrire une fonction de réparation afin de tenter de produire une solution faisable pour PCRDE à partir de la solution retournée par le problème maître à chaque itération. Si la valeur de la fonction objectif du problème maître est égale à la valeur correspondante de la meilleure solution réparée trouvée, alors la preuve de l'optimalité de cette dernière solution est acquise, et l'algorithme peut être arrêté sans qu'une solution admissible n'ait été trouvée par le problème maître. Une telle approche pourrait ainsi contribuer à accélérer la recherche tout en retournant une solution réalisable et une borne supérieure de M_{min} même lorsqu'aucune solution optimale n'a pu être trouvée dans le temps de calcul imparti. Ces idées sont mises en application au chapitre suivant.

Chapitre 4

Communication multicast pour les réseaux de fibre optique

Résumé

Ce chapitre aborde deux problèmes qui se posent lors du déploiement d'un réseau de fibre optique utilisant le multiplexage en longueur d'onde. L'objectif est de minimiser le coût des appareils opto-électroniques nécessaires à la communication multicast. Ces appareils coûteux doivent être installés partout où le signal doit être dupliqué, c'est-à-dire en chaque sommet de degré supérieur ou égal à trois dans le graphe représentant le réseau de fibre optique que l'on cherche à concevoir. Un sommet-branche est précisément un sommet dont le degré est supérieur ou égal à trois. Le premier problème abordé dans ce chapitre est noté MBV pour *Minimum Branch Vertex*, il consiste à trouver un arbre couvrant pour lequel le nombre de sommets-branche est minimum. Le second problème est noté MDS pour *Minimum Degree Sum*, il vise à trouver un arbre couvrant pour lequel la somme des degrés des sommets-branche est minimum. Une étude des relations entre MBV, MDS et le problème consistant à construire un arbre couvrant dont le nombre de feuilles est minimum est d'abord conduite, avant de proposer de nouvelles formulations à base de programmation linéaire en nombres entiers pour MBV et MDS, ainsi qu'un algorithme de plans coupants pour chacun d'eux. Une fonction de réparation est également proposée pour produire des solutions faisables à partir des solutions candidates retournées à chaque itération des algorithmes de plans coupants, ainsi qu'une recherche tabou destinée à améliorer ces solutions réparées. Les approches hybrides résultantes produisent des solutions de meilleure qualité que les plans coupants seuls, et que les heuristiques publiées dans [14].

4.1 Introduction

Ces dix dernières années ont vu le déploiement de réseaux de fibre optique augmenter massivement, accompagnant le développement de l'Internet [38]. En plus de la large bande passante qu'elle offre, la fibre optique peut tirer profit des dernières avancées technologiques en électronique et en optique sans nécessiter un remplacement complet. A titre d'exemple, le multiplexage en longueur d'onde permet d'augmenter la bande passante d'un réseau de fibre optique existant sans avoir à poser de nouvelles lignes de fibre optique, car seuls les multiplexeurs devront être remplacés [74].

Le multiplexage en longueur d'onde permet de faire transiter simultanément plusieurs signaux sur une même ligne de fibre optique, à condition que ces signaux exploitent des longueurs d'onde distinctes. Cette technologie est disponible depuis la fin des années 1970, et a suscité beaucoup d'intérêt depuis [66], en raison des avantages qu'elle offre en termes d'évolutivité et de bande passante.

Plus spécifiquement, l'exploitation du multiplexage en longueur d'onde à moindre coût peut être formulé comme suit. Etant donné un ensemble de chemins de communication (entre un émetteur et un récepteur), le nombre minimum de longueurs d'onde distinctes à utiliser dans le réseau est appelé *charge maximale d'un lien* [40], et sa valeur est égale au nombre maximum de communications différentes utilisant le même lien. En théorie, il est facile de limiter le nombre de longueurs d'ondes à ce niveau minimum sur tout le réseau, à condition d'installer un transpondeur (c'est-à-dire un convertisseurs de longueurs d'onde) en chaque sommet du graphe, ce qui représente un coût prohibitif car ces équipements demeurent chers [24]. Ainsi, il est classique de chercher à minimiser le nombre de transpondeurs tout en minimisant le nombre de longueurs d'onde utilisées dans le réseau [41]. Ce problème est $\mathcal{NP}$-difficile au sens fort (voir [70]).

Avec le développement de l'Internet et de la communication multicast [43], l'utilisation des réseaux de fibre optique évolue elle aussi. Il devient souhaitable que chaque sommet puisse diffuser les mêmes données à tous les autres sommets du réseau : la communication multicast répond ainsi aux besoins créés par la généralisation de la télévision haute définition, de la vidéo à la demande, et de la visio-conférence. Dans ce contexte, l'utilisation de transpondeurs pour convertir les longueurs d'onde utilisées est indispensable [31], et la mise en place à moindre coût de la communication multicast consiste alors à trouver un arbre couvrant tous les sommets du graphe, où chaque sommets-branche se voit affecter un transpondeur.

Plus formellement, soit G le graphe simple, non orienté représentant le réseau de fibre optique à équiper pour la communication multicast. Il est bien connu que déterminer si G admet une chaîne hamiltonienne est un problème $\mathcal{NP}$-complet. Si G est hamiltonien, alors cette chaîne hamiltonienne constitue un arbre sans sommet-branche et la communication multicast peut être mise en place sans ajouter de transpondeur, ce qui est le cas le plus favorable.

Ce chapitre aborde deux problèmes proches relatifs au déploiement du multiplexage en longueur d'onde dans le cadre d'une utilisation multicast, qui peuvent être vus comme des généralisations du problème visant à trouver une chaîne hamiltonienne dans un graphe. Le premier problème, noté MBV pour *Minimum Branch Vertex* consiste à trouver un arbre couvrant minimisant le nombre de sommets-branche [5, 31]. Cela revient à minimiser le nombre des transpondeurs, en faisant implicitement l'hypothèse que le coût de ces appareils est indépendant du degré des sommets branche. Le second problème, noté MDS, consiste à trouver un arbre couvrant minimisant la somme des

degrés de ses sommets-branche. Il a été introduit dans [14] comme une version plus réaliste de MBV, dans la mesure où le coût des appareils associés aux sommets-branche est supposé être proportionnel au degré de ces sommets. Bien sûr, pour un graphe hamiltonien, MBV et MDS ont une fonction objectif dont la valeur à l'optimum est nulle.

Il a été prouvé que MBV et MDS étaient $\mathcal{NP}$-difficiles au sens fort et difficiles à approximer dans [31] et [14] respectivement. On peut trouver des formulations à base de programmation linéaire en nombres entiers et des heuristiques pour ces deux problèmes dans [14]. On cherche ici à améliorer ces résultats en proposant des algorithmes combinant plans coupants et recherche tabou, afin de contourner l'inconvénient classique des algorithmes de plans coupants qui ne permettent pas d'obtenir de solution réalisable avant qu'une solution optimale ne soit trouvée. A cette fin, une fonction de réparation est appelée à l'issue de chaque itération afin de produire une solution faisable, qui est ensuite améliorée par une recherche tabou. Ainsi, même si aucune solution optimale n'a été trouvée au bout d'une durée maximale allouée à la recherche, l'algorithme proposé retourne une solution faisable trouvée ainsi que l'écart maximal de cette solution à l'optimum.

Ce chapitre est découpé comme suit. La section 4.2 met en évidence les relations entre MBV, MDS et le problème consistant à trouver un arbre couvrant dont le nombre de feuilles est minimum. L'algorithme de plans coupants pour aborder MBV est présenté en détail dans la section 4.3, puis il est combiné à une fonction de réparation et à une recherche tabou. La section 4.4 est dévolue à MDS. Comme ce problème est très proche de MBV, seules les différences avec les approches de la section précédentes sont présentées. Dans la section 4.5, les approches proposées sont comparées les unes aux autres, et avec les approches publiées dans [14]. La section 4.6 clôt le chapitre.

4.2 Propriétés

4.2.1 Minimisation du nombre de transpondeurs

En plus de MBV et de MDS, les auteurs de [14] mentionnent un troisième problème relatif à la conception des réseaux optiques. Tout sommet branche doit être équipé de répéteurs pour transmettre les signaux aux sommets voisins. Le nombre de répéteurs à installer sur un sommet-branche est égal à son degré moins deux. Le problème consistant à trouver un arbre couvrant minimisant le nombre de répéteurs est noté MSW. On montre maintenant que MSW et le problème consistant à trouver un arbre couvrant dont le nombre de feuilles est minimal, sont équivalents.

Lemme 4. *Trouver un arbre couvrant dont le nombre de répéteurs est minimum est équivalent à trouver un arbre couvrant dont le nombre de feuilles est minimum.*

Démonstration. Soit f_L le nombre de feuilles et f_{MSW} le nombre de répéteurs dans l'arbre couvrant T sur le graphe simple, non orienté G.

Pour tout sommet $v \in V$, le degré de v dans T peut être écrit comme suit :

$$\delta(v) = \min(2, \delta(v)) + \max(0, \delta(v) - 2)$$

Le second terme du second membre de l'égalité ci-dessus est le nombre de répéteurs requis au sommet v : ce nombre est nul si $\delta(v) \leq 2$. Si l'on additionne ces égalités pour tout v dans V, on obtient :

$$2(n-1) = \sum_{v \in V} \min(2, \delta(v)) + f_{MSW}$$

On remarque que $\min(2, \delta(v))$ est égal à 2 pour tous les sommets à l'exception des feuilles pour lesquelles cette quantité vaut 1. Comme il n'existe aucun sommet isolé dans un arbre couvrant, la somme sur V dans la dernière égalité peut être remplacée par $2n - f_L$, d'où :

$$f_L = 2 + f_{MSW}$$

□

On montre maintenant qu'il existe une relation étroite entre la somme des degrés des sommets-branche, notée f_{MDS}, le nombre de sommets-branche f_{MBV} et le nombre de feuilles d'un arbre couvrant.

Lemme 5. *L'égalité suivante est vraie pour tout arbre couvrant.*

$$f_{MDS} = 2f_{MBV} + f_L - 2$$

Démonstration. La somme des degrés de tous les sommets d'un arbre couvrant peut être séparée en deux termes comme suit

$$\sum_{v\in V}\delta(v) = 2(n-1) = \sum_{\substack{v\in V\\ \delta(v)\geq 3}}\delta(v) + \sum_{\substack{v\in V\\ \delta(v)\leq 2}}\delta(v) = f_{MDS} + \sum_{\substack{v\in V\\ \delta(v)\leq 2}}\delta(v)$$

Le dernier terme de la somme s'écrit comme suit

$$\sum_{\substack{v\in V\\ \delta(v)\leq 2}}\delta(v) = 2n - 2\sum_{v\in V} b_v - \sum_{\substack{v\in V\\ \delta(v)=1}}\delta(v)$$

Où b_v est une variable binaire égale à 1 si et seulement si v est un sommet-branche. En effet, le premier membre est égal à deux fois le nombre de sommets moins la contribution des sommets-branche à cette somme, moins la contribution des feuilles.

En remplaçant $\sum_{\substack{v\in V\\ \delta(v)\leq 2}}\delta(v)$ dans l'expression de la somme des degrés, on a

$$2(n-1) = f_{MDS} + 2n - 2f_{MBV} - f_L$$

□

Le lemme 5 suggère que MBV, MDS et MSW sont des problèmes très proches, car ils partagent les mêmes contraintes et ne diffèrent que par leurs fonctions objectif, qui sont reliées par une égalité. Cela soulève la question de leur indépendance, qui fait l'objet de la sous-section suivante.

4.2.2 Indépendance de MBV, MDS et MSW

Si une chaîne hamiltonienne existe, elle constitue une solution optimale aux trois problèmes. De plus, les premiers tests numériques pratiqués sur ces problèmes ont suggéré que toute solution optimale de MDS était aussi optimale pour MBV. Deux problèmes $\mathcal{P}_1$ et $\mathcal{P}_2$ sont dits équivalents si et seulement si toute solution optimale pour $\mathcal{P}_1$ est également optimale pour $\mathcal{P}_2$ et réciproquement. En plus de l'équivalence, on peut définir la notion de dominance comme suit : le problème $\mathcal{P}_1$ domine le problème $\mathcal{P}_2$ si toute solution optimale pour $\mathcal{P}_1$ est également optimale pour $\mathcal{P}_2$ (lorsque la réciproque est vraie, $\mathcal{P}_1$ et $\mathcal{P}_2$ sont équivalents). Un exemple bien connu de la relation de dominance est celui des deux problèmes d'ordonnancement non préemptifs à une machine noté $1||L_{max}$ et $1||T_{max}$ dans le système de notation à trois champs [8, 34]. On rappelle que le retard algébrique maximum L_{max} et le retard absolu maximum T_{max} d'un ensemble de n opérations sont définis par

$$L_{max} = \max_{j\in\{1,\ldots,n\}}(c_j - d_j), \qquad T_{max} = \max(0, L_{max})$$

Où d_j désigne la date échue de l'opération j et c_j est la date de fin de l'opération j, pour tout j dans $\{1,\ldots,n\}$. Si une solution est optimale pour $1||L_{max}$ avec $L_{max} \geq 0$, alors elle est aussi optimale pour $1||T_{max}$ car $T_{max} = \max(0, L_{max}) = L_{max}$ dans ce cas. Si la solution optimale à $1||L_{max}$ est telle que $L_{max} < 0$, alors $T_{max} = 0$ et cette solution est également optimale pour $1||T_{max}$ car $T_{max} \geq 0$ par définition. Ainsi, $1||L_{max}$ domine $1||T_{max}$ mais les deux problèmes ne sont pas équivalents comme on peut le montrer avec un contre-exemple de la réciproque choisi de façon à ce que L_{max} soit strictement négatif à l'optimum. On peut déduire de ce résultat de dominance que l'algorithme EDD (pour *Earliest Due Date*) [51] qui construit une solution optimale pour $1||L_{max}$ peut être utilisé pour résoudre $1||T_{max}$, et par conséquent il n'y a pas lieu de chercher un algorithme spécifiquement dédié à deux problèmes de même complexité computationnelle si l'un domine l'autre. Si deux problèmes ne sont pas équivalents et qu'il n'existe aucune relation d'équivalence entre eux, il sont indépendants, et le développement d'un algorithme pour chacun d'entre eux est pleinement justifié. Dans cette section, on exhibe deux contre-exemples pour démontrer qu'il n'existe aucune relation d'équivalence entre MBV, MDS et MSW.

La figure 4.1 illustre et nomme les six relations de dominance possibles entre les trois problèmes.

Premièrement, on montre que MBV et MSW sont indépendants à l'aide du graphe G' de la figure 4.2.

La solution $\mathcal{T}_1'$ de la figure 4.3 est optimale pour MDS et pour MBV sur G'. La solution $\mathcal{T}_2'$ de la figure 4.4 est optimale pour MSW sur G'. Les sommets-branche sont colorés en gris sur ces deux figures.

Lorsqu'une solution est optimale pour un problème, la valeur de l'objectif correspondante est reportée en caractères gras dans la table 4.1. Cette table montre que les relations de dominance D_3, D_4, D_5 et D_6 ne sont pas vraies, ce qui implique que MSW n'est équivalent ni à MBV, ni à MDS.

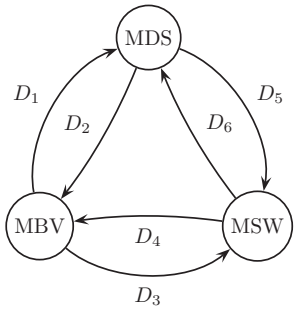


FIGURE 4.1 – Les six relations de dominances possibles entre MBV, MDS et MSW.

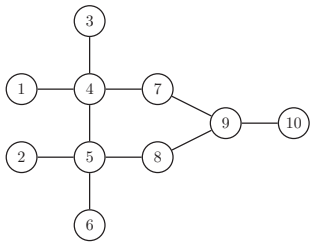


FIGURE 4.2 – Graphe G' servant de premier contre-exemple.

Deuxièmement, on montre que MBV et MDS sont indépendants. Un contre-exemple fourni dans [14] permet de montrer que la relation de dominance D_1 n'est pas vraie. On produit un autre contre-exemple pour montrer que toute solution optimale à MDS n'est pas nécessairement optimale pour MBV.

La table 4.2 montre que toute solution optimale pour MDS ne l'est pas nécessairement pour MBV.

Il résulte de cette étude que MBV, MDS et MSW sont trois problèmes indépendants. Il est donc justifié de développer des algorithmes spécifiques pour chacun d'eux. Le problème consistant à minimiser le nombre de feuilles d'un arbre couvrant est bien connu, et ne sera pas étudié ici : des approches exactes et des schémas d'approximation pour MSW peuvent être trouvés dans [26, 62]. Le reste de ce chapitre est consacré au développement d'approches basées sur les plans coupants pour résoudre MBV et MDS.

4.3 Méthodes de résolution de MBV

La section 4.3.1 propose un algorithme de plans coupants s'appuyant sur la génération de deux types de coupes. Une fonction de réparation est introduite dans la section 4.3.2 afin de construire des solutions faisables pour MBV à partir des résultats retournés par l'algorithme de plans coupants à chaque itération. La section 4.3.3 se focalise sur la génération de coupes à faible coût computationnel, réalisée en allouant une durée maximale variable à la résolution du problème maître à chaque itération, et par l'ajout d'une nouvelle contrainte sur la valeur de l'objectif exploitant l'objectif de la meilleure solution faisable disponible. Une première version de l'algorithme de plans coupants, notée CPA_{MBV}, est dotée de ces caractéristiques. Une seconde version notée CPA^{TS}_{MBV}, est construite en y ajoutant une recherche tabou visant à améliorer les solutions retournées par la fonction de réparation. Cette recherche tabou est présentée dans la section 4.3.4.

Ce type d'algorithme de plans coupants a été utilisé avec succès dans [46] pour résoudre le *minimum power symmetry connectivity problem*, et dans [44] pour le *minimum power multicasting problem in wireless networks*. MBV et MDS partagent des caractéristiques communes avec les problèmes abordés dans [46] et [44] : la fonction objectif ne fait référence qu'à des quantités attachées aux sommets, alors que des contraintes difficiles (c'est-à-dire relatives à la connexité) doivent être vérifiées par les arêtes. Les algorithmes présentés ici se fondent sur le même type d'approches mais tirent parti de solutions réalisables pour accélérer la convergence.

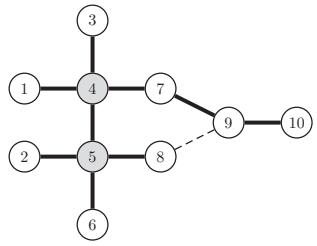


FIGURE 4.3 – $\mathcal{T}_1'$, solution optimale pour MBV et MDS sur G'.

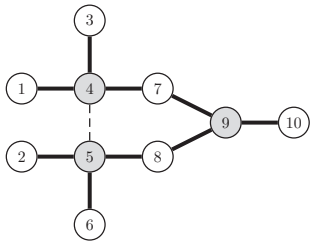


FIGURE 4.4 – $\mathcal{T}_2'$, solution optimale pour MSW sur G'.

TABLE 4.1 – Valeurs de l'objectif atteintes par $\mathcal{T}_1'$ et $\mathcal{T}_2'$ sur G' pour MBV, MDS et MSW.

Solution	$\mathcal{T}_1'$	$\mathcal{T}_2'$
Nombre de sommets-branche (MBV)	**2**	3
Somme des degrés des sommets-branche (MDS)	**8**	9
Nombre de répéteurs (MSW)	4	**3**

4.3.1 Algorithme de plans coupants

Une formulation par programmation linéaire en nombres entiers pour MBV a été proposée dans [14]. L'algorithme de plans coupants proposé pour aborder MBV sert de base à CPA_{MBV} et à CPA^{TS}_{MBV}. Il fait intervenir deux types de variables de décision binaires : pour tout $e \in E$, x_e prend la valeur 1 si l'arête e fait partie de la solution ($x_e = 0$ sinon). Pour tout v dans V, y_v prend la valeur 1 si le sommet v est un sommet-branche ($y_v = 0$ sinon). On notera $I(v)$ l'ensemble des arêtes incidentes au sommet v, et le degré de v dans G est alors défini par $\delta_G(v) = |I(v)| \geq 1$ pour tout $v \in V$.

MBV peut être formulé par le programme linéaire en nombres entiers noté IP_{MBV} et défini par :

$$\text{Minimiser} \quad \sum_{v \in V} y_v \tag{4.1}$$

$$\text{Sous les contraintes} \quad \sum_{e \in I(v)} x_e \geq 1 \qquad \forall v \in V \tag{4.2}$$

$$\sum_{e \in E} x_e = n - 1 \tag{4.3}$$

$$y_v = 0 \qquad \forall v \in V, |I(v)| \leq 2 \tag{4.4}$$

$$\sum_{e \in I(v)} x_e - 2 \leq (|I(v)| - 2)\, y_v \qquad \forall v \in V \tag{4.5}$$

$$\sum_{e \in CB(S)} x_e \geq 1 \qquad \forall S \subset V \tag{4.6}$$

$$x_e \in \{0, 1\} \qquad \forall e \in E \tag{4.7}$$

$$y_v \in \{0, 1\} \qquad \forall v \in V \tag{4.8}$$

(4.1) est la fonction objectif. (4.2) interdit l'apparition de sommets isolés dans la solution. (4.3) assure que la solution est constituée de $n - 1$ arêtes. Les contraintes (4.4) et (4.5) assurent que y_v est fixé à 1 dès lors que le degré du sommet v dans la solution est supérieur ou égal à trois. (4.6) est l'ensemble de toutes les contraintes de connexité, qui assurent que la solution est un sous-graphe de G connecté. Plus précisément, S est un sous-ensemble strict quelconque de V, et $CB(S)$ est le *coboundary* de S [48], c'est-à-dire l'ensemble de toutes les arêtes ayant une et une seule extrémité dans S. Le nombre de ces contraintes croît généralement de façon exponentielle avec la taille de G, c'est pour cela que IP_{MBV} ne peut être résolu que pour des graphes de taille très modeste.

Par conséquent on construit le problème maître, qui est initialement constitué des équations (4.1)−(4.5) et

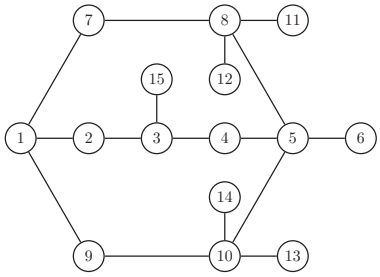


FIGURE 4.5 – Graphe G'' servant de second contre-exemple.

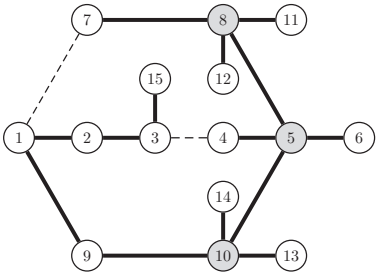


FIGURE 4.6 – $\mathcal{T}_1''$, solution optimale pour MBV sur G''.

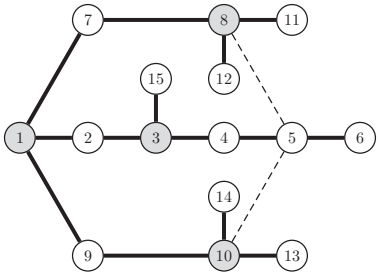


FIGURE 4.7 – $\mathcal{T}_2''$, solution optimale pour MDS sur G''.

(4.7)−(4.8), il s'agit donc d'une relaxation de IP_{MBV}, dans laquelle toutes les contraintes de connexité ont été ignorées. Une solution du problème maître est appelé un *candidat arbre* car l'ensemble d'arêtes correspondant peut avoir des cycles, et ne pas être connexe.

Après avoir résolu le problème maître, on teste la connexité du candidat arbre. S'il n'est pas connexe, alors des coupes sont introduites dans le problème maître comme suit. Remarquons d'abord que les contraintes de connexité de IP_{MBV} peuvent être remplacées par les contraintes de cycle suivantes :

$$\sum_{e \in CY} x_e \leq |CY| - 1 \qquad \forall CY \in \mathcal{C}_G \tag{4.9}$$

Où $\mathcal{C}_G$ est l'ensemble de tous les cycles de G, et où CY désigne un cycle. Là aussi, comme le nombre de ces contraintes croît de manière exponentielle avec la taille de G, l'algorithme de plans coupants peut aussi introduire des coupes de cycle dans le problème maître si c'est nécessaire, c'est-à-dire si le candidat arbre retourné par le problème maître a un ou plusieurs cycles.

Plus précisément, le nombre de cycles ν d'un graphe est encadré comme suit [68] :

$$\mu \leq \nu \leq 2^{\mu} - 1$$

Où μ est le nombre cyclomatique de G, c'est-à-dire le nombre minimum d'arêtes à retirer pour que G soit sans cycle. Il est défini par

$$\mu = m - n + \kappa$$

Où κ est le nombre de composantes connexes de G. Comme toute solution du problème maître est constituée de $n - 1$ arêtes, $\mu = \kappa - 1$, donc le nombre de cycles dans tout candidat arbre est encadré comme suit.

$$\kappa - 1 \leq \nu \leq 2^{\kappa - 1} - 1$$

On choisit d'énumérer $\kappa - 1$ cycles dans le candidat arbre afin de générer le même nombre de coupes de cycle, tout en évitant d'introduire des coupes *denses* [22] dans le problème maître. On recherche donc des cycles de cardinalité

TABLE 4.2 – Valeurs de l'objectif atteintes par $\mathcal{T}_1''$ et $\mathcal{T}_2''$ sur G'' pour MBV et MDS.

Solution	$\mathcal{T}_1''$	$\mathcal{T}_2''$
Nombre de sommets-branche (MBV)	**3**	4
Somme des degrés des sommets-branche (MDS)	12	**12**

minimale sur le candidat arbre, en recourant à une heuristique simple basée sur un calcul de plus court chemin, présentée dans l'algorithme 5.

Bien qu'un algorithme de plans coupants générant uniquement des coupes de connexité (ou uniquement des coupes de cycle) soit en théorie suffisant pour converger vers une solution optimale de MBV, la génération conjointe de ces deux types de coupes implémentée dans CPA_{MBV} et CPA^{TS}_{MBV} est beaucoup plus efficace en pratique. Les figures 4.8 et 4.9 présentent deux exemples qui illustrent l'intérêt de cette génération conjointe.

L'algorithme de plans coupants ne produisant que des coupes de cycles est d'abord exécuté sur l'instance de la figure 4.8. Comme on le voit sur la figure 4.10, deux cycles affectent la solution optimale du problème maître. Cependant, la génération de coupes de cycles peut se révéler inefficace car jusqu'à $n-2$ paires de cycles peuvent affecter les solutions optimales au problème maître avant qu'une solution optimale (et donc réalisable) pour MBV ne soit trouvée. Plus formellement, pour tout $p \in \{1, \ldots, n-2\}$, la paire de cycles p est définie par $(\{e_1, \ldots, e_p, e_{2n+1}, e_{2n+p+1}, e_{n+1}, \ldots, e_{n+p}\}, \{e_{p+2}, \ldots, e_n, e_{2n+p+2}, e_{3n+1}, e_{n+p+2}, \ldots, e_{2n}\})$. La figure 4.10 montre la paire de cycles associée à $p = 1$. On observe que la coupe de connexité associée à la composante connexe constituée des extrémités des arêtes e_{3n+3} et e_{3n+4} tend à faire apparaître un sommet-branche, que le problème maître s'efforce d'éviter en créant des cycles. Cela montre qu'en générant une coupe de connexité dans cet exemple, on peut faire l'économie de $n-2$ itérations par comparaison avec un algorithme de plans coupants qui ne produirait que des coupes de cycle.

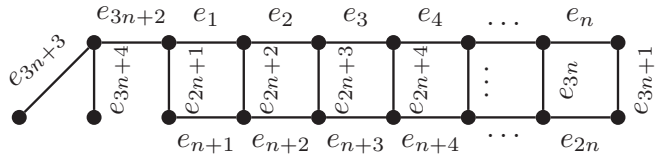


FIGURE 4.8 – Graphe illustrant l'usage exclusif de coupes de cycles.

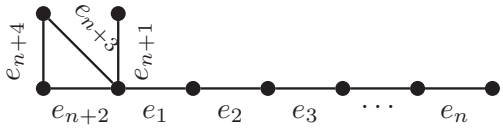


FIGURE 4.9 – Graphe illustrant l'usage exclusif de coupes de connexité.

L'algorithme de plans coupants ne produisant que des coupes de connexité est maintenant exécuté sur l'instance de la figure 4.9. Comme on le voit sur la figure 4.11, deux composantes connexes affectent la solution optimale au problème maître. Cependant, la génération de coupes de connexité peut se révéler inefficace car jusqu'à $n-2$ paires de composantes connexes peuvent affecter les solutions optimales au problème maître avant qu'une solution optimale (et donc réalisable) pour MBV ne soit trouvée. Plus formellement, pour tout $p \in \{1, \ldots, n-2\}$, la paire de composantes connexes p est définie par le graphe résultant de la suppression de l'arête e_{p+1}. La figure 4.11 montre la paire de composantes connexes correspondant à $p = 1$. On observe que la coupe de cycle associée à $\{e_{n+2}, e_{n+3}, e_{n+4}\}$ conduit immédiatement à la solution optimale. Cela montre qu'on générant une coupe de cycle dans cet exemple, on peut faire l'économie de $n-2$ itérations par comparaison avec un algorithme de plans coupants qui ne produirait que des coupes de connexité.

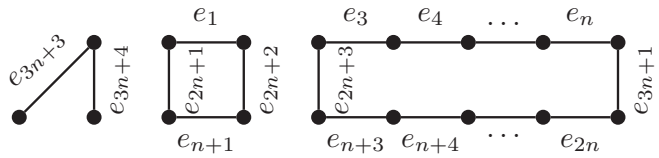


FIGURE 4.10 – Une des $n-2$ paires de cycles.

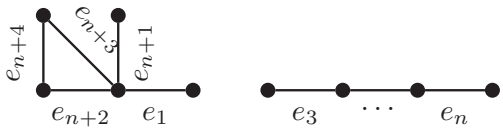


FIGURE 4.11 – Une des $n-2$ paires de composantes connexes.

L'algorithme de plans coupants est décrit plus en détail dans l'algorithme 4. Le nombre de sommets-branche de la meilleure solution faisable est noté f_{UB}, et est initialisé à n, de sorte qu'il sera mis à jour dès que la première solution faisable pour MBV aura été obtenue. La boucle principale de l'algorithme de plans coupants est exécutée jusqu'à ce qu'une solution optimale soit trouvée (le booléen *stop* est alors fixé à `true`). Le problème maître est résolu par un solver, et retourne le candidat arbre $\mathcal{T}$, qui est un ensemble de $n-1$ arêtes couvrant le graphe G, et la valeur de l'objectif correspondant est notée f. On recherche ensuite les cycles et les composantes connexes de ce candidat arbre à l'aide de la fonction $CCCD$ (pour *Cycles and Connected Components Detection*). Cette fonction est présentée en détail dans l'algorithme 5.

$CCCD$ retourne les quatre sorties suivantes.

- κ, le nombre de composantes connexes de $\mathcal{T}$,
- $(S_k)_{k\in\{1,\dots,\kappa\}}$, l'ensemble des κ composantes connexes,
- $(CB_k)_{k\in\{1,\dots,\kappa-1\}}$, l'ensemble des $\kappa-1$ coboundaries,
- $(CY_j)_{j\in\{1,\dots,\kappa-1\}}$, l'ensemble des $\kappa-1$ cycles distincts trouvés dans $\mathcal{T}$.

Si $\mathcal{T}$ est connexe, alors c'est un arbre couvrant, c'est-à-dire que $\mathcal{T}$ est une solution optimale pour MBV, et l'algorithme se termine. Si $\mathcal{T}$ n'est pas connexe, la fonction de réparation $Repair_{MBV}$ est appelée afin de construire une solution faisable $\mathcal{T}_k$ à partir de $\mathcal{T}$, en partant de la composante connexe S_k, pour tout k dans $\{1,\dots,\kappa\}$. La fonction $Repair_{MBV}$ est décrite en détail dans l'algorithme 6. La meilleure solution trouvée depuis le début de la recherche, notée $\mathcal{T}_{UB}$ et la valeur de l'objectif qui lui est associée (notée f_{UB}) sont remplacés par $\mathcal{T}_k$ et f_k si $f_k < f_{UB}$. Si f_{UB} est égale à la valeur de l'objectif du problème maître, alors l'optimalité de $\mathcal{T}_{UB}$ est démontrée et l'algorithme retourne cette solution ainsi que la valeur de l'objectif qui lui est associée.

Enfin, $\kappa-1$ coupes de connexité et $\kappa-1$ coupes de cycle sont ajoutées au problème maître, qui sera résolu de nouveau jusqu'à ce qu'une solution optimale pour MBV soit trouvée.

Etant donnée une solution $\mathcal{T}$ au problème maître, la fonction $CCCD$ (voir l'algorithme 5) initialise le nombre de composantes connexes, κ, à un, le nombre de cycles, σ, à zéro, et l'ensemble des sommets à explorer, V_S, à V. Cette fonction construit S_κ, la composante connexe courante de $\mathcal{T}$. Un sommet v_0 est choisi arbitrairement dans V_S pour initialiser S_κ. Le choix de v_0 est sans importance car toutes les composantes connexes de $\mathcal{T}$ seront énumérées. L'ensemble des arêtes explorées de $\mathcal{T}$, noté E_S, est initialement vide. Le booléen *stable* est fixé à `false` tant que la composante connexe courante S_κ peut être étendue. Si l'on trouve une arête dans $\mathcal{T}\backslash E_S$ dont une et une seule extrémité appartient à S_κ, alors son autre extrémité est ajoutée à S_κ, et l'arête en question est ajoutée à E_S. Si l'on trouve une arête dans $\mathcal{T}\backslash E_S$ dont les deux extrémités sont dans S_κ, alors cela signifie qu'il existe un cycle dans E_S. Le cycle de longueur minimale contenant cette arête est obtenu en calculant le plus court chemin entre les deux extrémités de cette arête dans le graphe $G_S = (S_\kappa, E_S)$, où toutes les arêtes de G_S ont un poids unitaire. Le compteur de cycles, σ, est alors incrémenté de un et le cycle trouvé est sauvegardé dans CY_σ. Enfin, l'arête dont les deux extrémités sont dans S_κ est ajoutée à E_S.

Toutes les arêtes de $\mathcal{T}\backslash E_S$ sont ainsi testées jusqu'à ce que le booléen *stable* soit égal à `true`. V_S est alors remplacé par $V_S\backslash S_\kappa$. Si V_S est vide, alors la fonction retourne ses résultats et se termine. Dans le cas contraire, les sommets de S_κ ne sont pas connectés à ceux de $V_S\backslash S_\kappa$, et une contrainte de connexité doit être ajoutée au problème maître. A cette fin, le coboundary de S_κ, CB_κ, est défini par l'ensemble des arêtes de E ayant une et une seule extrémité dans S_κ. Comme au moins une autre composante connexe existe dans ce cas, κ est incrémenté de un et S_κ est à nouveau initialisé avec un sommet arbitrairement choisi dans V_S. Le processus se répète jusqu'à ce que V_S soit vide, puis $CCCD$ retourne κ, $(S_k)_{k\in\{1,\dots,\kappa\}}$, $(CB_k)_{k\in\{1,\dots,\kappa-1\}}$, et $(CY_j)_{j\in\{1,\dots,\kappa-1\}}$.

Le fait que la fonction $CCCD$ décrite par l'algorithme 5 retourne exactement $\kappa-1$ cycles différents n'est pas nécessairement évident. Ce résultat est prouvé ci-après.

Lemme 6. *$CCCD$ retourne $\kappa-1$ cycles différents.*

Démonstration. L'entrée de la fonction $CCCD$ est un candidat arbre $\mathcal{T}$, c'est-à-dire un graphe constitué de n sommets et de $n-1$ arêtes. Ce graphe a κ composantes connexes, donc $\kappa-1$ arêtes manquent pour connecter ce graphe. Dans $\mathcal{T}$, ces $\kappa-1$ arêtes relient des paires de sommets appartenant à la même composante connexe, où elles sont responsables de l'apparition de cycles. Pour tout k dans $\{1,\dots,\kappa\}$, soit S_k une composante connexe de $\mathcal{T}$, et soit μ_k le nombre cyclomatique de S_k. Par définition, $\mu_k = m_k - n_k$ où m_k est le nombre d'arêtes de $\mathcal{T}$ dont les deux extrémités appartiennent à S_k, et $n_k = |S_k|$. La somme des μ_k pour tout k dans $\{1,\dots,\kappa\}$ est égale à $\kappa-1$, puisque toute arête qui fait défaut pour connecter $\mathcal{T}$ correspond à une arête à l'origine d'un cycle dans une des composantes connexes de $\mathcal{T}$. Comme les composantes connexes satisfaisant $\mu_k = 0$ sont sans cycle, on ne considère que les composantes connexes S_k pour lesquelles $\mu_k \geq 1$. Lors de la construction de S_k (lignes 9 à 23 dans l'algorithme 5), chaque arête dont au moins une extrémité appartient à S_k est utilisée une et une seule fois, soit pour connecter un nouveau sommet à S_k (lignes 12 à 15), soit non parce que les deux extrémités de cette arête

```
   /* Initialisation                                                                    */
 1 Le problème maître est initialement défini par les équations (4.1) − (4.5) and (4.7) − (4.8) ;
 2 stop ⟵ false;
 3 f_UB ⟵ n;
 4 while stop = false do
 5     (T, f) ⟵ Solution optimale du problème maître ;
       /* Recherche des cycles et des composantes connexes de T                        */
 6     (κ, (S_k)_{k∈{1,...,κ}}, (CB_k)_{k∈{1,...,κ−1}}, (CY_j)_{j∈{1,...,κ−1}}) ⟵ CCCD(T);
 7     if κ = 1 then
           /* T est un arbre couvrant                                                  */
 8         T_UB ⟵ T;
 9         f_UB ⟵ f;
10         stop ⟵ true;
11     else
           /* T n'est pas un arbre                                                     */
           /* Réparation de T                                                          */
12         k ⟵ 1;
13         while k < κ and stop = false do
14             (T_k, f_k) ⟵ Repair_MBV(T, S_k);
15             if f_k < f_UB then
16                 T_UB ⟵ T_k;
17                 f_UB ⟵ f_k;
18             end
19             if f_UB = f then
20                 stop ⟵ true;
21             end
22             k ⟵ k + 1;
23         end
           /* Ajouts des coupes au problème maître                                     */
24         Ajouter les coupes de connexité suivantes au problème maître ;
25         Σ_{e∈CB_k} x_e ≥ 1    ∀k ∈ {1, ..., κ − 1};
26         Ajouter les coupes de cycle suivantes au problème maître ;
27         Σ_{e∈CY_j} x_e ≤ |CY_j| − 1    ∀j ∈ {1, ..., κ − 1};
28     end
29 end
30 return (T_UB, f_UB);
```

Algorithme 4: Algorithme de plans coupants pour MBV.

appartiennent déjà à S_k (lignes 16 à 20). Puisque S_k est une composante connexe en cours de construction, $n_k - 1$ arêtes seront utilisées pour l'agrandir en atteignant de nouveaux sommets, et μ_k autres arêtes seront utilisées pour énumérer un cycle. On peut aisément vérifier que chaque arête e dans cet ensemble de μ_k arêtes est utilisée pour construire un cycle de S_k, qui est distinct de tous les cycles construits précédemment. En effet, chaque fois qu'un cycle est construit, il contient l'arête e, qui est alors utilisée pour la première fois. Cela est garanti par le fait qu'à la ligne 11, seules les arêtes non explorées interviennent pour construire S_k. Il en résulte que toute composante connexe S_k pour laquelle $\mu_k \geq 1$ est à l'origine de la construction de μ_k cycles distincts. Comme la somme des μ_k pour tout $k \in \{1, \ldots, \kappa\}$ est égale à $\kappa - 1$, la fonction $CCCD$ retourne bien $\kappa - 1$ cycles distincts. □

4.3.2 Fonction de réparation

La fonction de réparation $Repair_{MBV}$ est appelée à chaque itération de l'algorithme de plans coupants pour produire une solution faisable $\mathcal{T}_k$ et calculer son objectif f_k à partir du candidat arbre courant, et de la composante connexe S_k de celui-ci. Cette fonction est décrite en détail par l'algorithme 6 et se fonde sur l'idée suivante. Si $\mathcal{T}$ a

Input : $\mathcal{T}$ la solution du problème maître
`/* Initialisation */`
$\kappa \longleftarrow 1$;
$\sigma \longleftarrow 0$;
$V_S \longleftarrow V$;
while $|V_S| > 0$ **do**
  `/* Construction de` S_κ`, composante connexe de` V_S `*/`
  Soit v_0 un sommet quelconque de V_S ;
  $S_\kappa \longleftarrow \{v_0\}$;
  $E_S \longleftarrow \{\emptyset\}$;
  stable $\longleftarrow$ `false` ;
  while *stable* = *false* **do**
    stable $\longleftarrow$ `true` ;
    for $(u, v) \in \mathcal{T} \backslash E_S$ **do**
      if $u \in S_\kappa$ `and` $v \notin S_\kappa$ **then**
        $S_\kappa \longleftarrow S_\kappa \cup \{v\}$;
        $E_S \longleftarrow E_S \cup \{(u, v)\}$;
        stable $\longleftarrow$ `false` ;
      else if $u \in S_\kappa$ `and` $v \in S_\kappa$ **then**
        `/* Un cycle est détecté dans` S_κ `*/`
        Calculer $\mathcal{P}$, le plus court chemin de u à v dans le graphe $G_S = (S_\kappa, E_S)$;
        $\sigma \longleftarrow \sigma + 1$;
        $CY_\sigma \longleftarrow \mathcal{P} \cup \{(u, v)\}$;
        $E_S \longleftarrow E_S \cup \{(u, v)\}$;
      end
    end
  end
  if $|S_\kappa| = n$ **then**
    `/* La solution` $\mathcal{T}$ `est connexe, pas de coboundary */`
    $CB_\kappa \longleftarrow \{\emptyset\}$;
  else
    `/* Une coupe de connexité est requise pour connecter` S_κ `à` $V_S \backslash S_\kappa$ `*/`
    `/* Calcul de` CB_κ `*/`
    $CB_\kappa \longleftarrow \{\emptyset\}$;
    for $(u, v) \in E \backslash E_S$ **do**
      if $u \in S_\kappa$ `and` $v \notin S_\kappa$ **then**
        $CB_\kappa \longleftarrow CB_\kappa \cup \{(u, v)\}$;
      end
    end
    $\kappa \longleftarrow \kappa + 1$;
  end
  $V_S \longleftarrow V_S \backslash S_\kappa$;
end
return $\left(\kappa, (S_k)_{k\in\{1,\ldots,\kappa\}}, (CB_k)_{k\in\{1,\ldots,\kappa-1\}}, (CY_j)_{j\in\{1,\ldots,\kappa-1\}}\right)$;

Algorithme 5: *CCCD* : Détection des cycles et des composantes connexes d'un candidat arbre $\mathcal{T}$.

$\kappa > 1$ composantes connexes, alors une solution faisable pour MBV peut être obtenue à partir de $\mathcal{T}$ en y ajoutant $\kappa - 1$ arêtes afin d'assurer la connexité de $\mathcal{T}_k$ et en en retirant $\kappa - 1$ arêtes pour en supprimer tous les cycles. Le choix des arêtes à ajouter et à retirer est fait à l'aide d'un algorithme glouton visant à diminuer autant que possible le nombre de sommets-branche lors du retrait d'une arête, et à limiter autant que possible d'augmentation du nombre de sommets-branche lors de l'ajout d'une arête.

Plus précisément, $\delta(v)$ est initialisé à 0 pour tout v dans V, et $\mathcal{T}_k$ est initialisé à l'ensemble vide. On prend un sommet arbitraire v_0 de S_k pour initialiser S, qui constitue l'ensemble des sommets couverts par l'arbre couvrant $\mathcal{T}_k$ en cours de construction. Comme dans la fonction *CCCD*, V_S est initialisé à V et constitue l'ensemble des sommets

explorés ; l'algorithme effectue une nouvelle itération tant que V_S est non vide. Le booléen *stable* est fixé à `false` tant que S peut être étendu. Si une arête de $\mathcal{T}\backslash\mathcal{T}_k$ a une et une seule extrémité dans S, alors son autre extrémité est ajoutée à S, l'arête en question est ajoutée à $\mathcal{T}_k$, et le degré de ses extrémités est incrémenté de un. Si une arête de $\mathcal{T}\backslash\mathcal{T}_k$ a ses deux extrémités dans S, alors il existe un cycle dans $\mathcal{T}_k$. Ce cycle est noté CY, il est calculé comme dans la fonction $CCCD$. L'arête notée $(u_{rem}, v_{rem}) \in CY$ est celle qui sera retirée de $\mathcal{T}_k$ pour supprimer ce cycle. Cette arête est choisie de sorte que son retrait provoque la plus forte diminution du nombre de sommets-branche. Plus précisément, si l'extrémité d'une telle arête est de degré trois, alors son retrait aura pour effet de diminuer le nombre de sommets-branche de un. Une fois l'arête (u_{rem}, v_{rem}) sélectionnée, elle est retirée de $\mathcal{T}_k$ et le degré de ses extrémités est décrémenté de un.

Une fois que S est stable, V_S est remplacé par $V_S\backslash S$. Si V_S est vide, la solution est faisable et la fonction retourne $\mathcal{T}_k$ et la valeur de son objectif f_k. Si V_S n'est pas vide, une arête n'appartenant pas à $\mathcal{T}$ doit être ajoutée à $\mathcal{T}_k$ pour connecter cet ensemble aux sommets de V_S. L'arête à ajouter est notée (u_{add}, v_{add}), elle est choisie parmi les arêtes ayant une et une seule extrémité dans S, de manière à ce que son introduction limite autant que possible l'augmentation du nombre de sommets-branche de $\mathcal{T}_k$. Plus précisément, si l'extrémité d'une telle arête est de degré deux, alors son introduction donnera naissance à un nouveau sommet-branche. Une fois l'arête (u_{add}, v_{add}) sélectionnée, elle est ajoutée à $\mathcal{T}_k$ et le degré de ses extrémités est incrémenté de un.

Cette fonction de réparation est appelée κ fois à chaque itération de l'algorithme de plans coupants. Son second argument S_k permet d'initialiser S avec un point quelconque de S_k dans une composante connexe de $\mathcal{T}$ différente à chaque appel afin de produire une solution faisable pour MBV potentiellement différente des autres solutions retournées pour différentes valeurs de k. Cette fonction de réparation peut s'avérer très utile pour faire diminuer le nombre d'itérations nécessaires à l'algorithme de plans coupants car une solution optimale peut ainsi être trouvée rapidement, notamment dans les graphes denses qui sont hamiltoniens dès lors que la condition de densité de Dirac [23] est satisfaite.

Input : $\mathcal{T}$ la solution du problème maître, et S_k, une composante connexe de $\mathcal{T}$
/* Initialisation */
$\delta(v) \longleftarrow 0 \quad \forall v \in V$;
Soit v_0 un sommet quelconque de S_k;
$S \longleftarrow \{v_0\}$;
$\mathcal{T}_k \longleftarrow \{\emptyset\}$; $V_S \longleftarrow V$;
while $|V_S| > 0$ **do**
  /* Construction de S */
  stable $\longleftarrow$ `false` ;
  while *stable* = *false* **do**
    stable $\longleftarrow$ `true` ;
    for $(u, v) \in \mathcal{T} \backslash \mathcal{T}_k$ **do**
      if $u \in S$ `and` $v \notin S$ **then**
        $S \longleftarrow S \cup \{v\}$;
        $\mathcal{T}_k \longleftarrow \mathcal{T}_k \cup \{(u, v)\}$; $\delta(u) \longleftarrow \delta(u) + 1$; $\delta(v) \longleftarrow \delta(v) + 1$;
        stable $\longleftarrow$ `false` ;
      else if $u \in S$ `and` $v \in S$ **then**
        /* Détection d'un cycle */
        Calculer $\mathcal{P}$, un plus court chemin de u à v dans le graphe $G_S = (S, \mathcal{T}_k)$;
        $CY \longleftarrow \mathcal{P} \cup \{(u, v)\}$;
        /* Sélection de la meilleure arête (localement) à retirer de $\mathcal{T}_k$ */
        $(u_{rem}, v_{rem}) \longleftarrow (0, 0)$; $dec\_max \longleftarrow 0$;
        for $(u', v') \in CY$ **do**
          $dec \longleftarrow 0$;
          if $\delta(u') = 3$ **then** $dec \longleftarrow dec + 1$;
          if $\delta(v') = 3$ **then** $dec \longleftarrow dec + 1$;
          if $dec > dec\_max$ **then** $dec\_max \longleftarrow dec$; $(u_{rem}, v_{rem}) \longleftarrow (u', v')$;
        end
        /* Retrait de l'arête (u_{rem}, v_{rem}) de $\mathcal{T}_k$ */
        $\mathcal{T}_k \longleftarrow \mathcal{T}_k \backslash \{(u_{rem}, v_{rem})\}$; $\delta(u_{rem}) \longleftarrow \delta(u_{rem}) - 1$; $\delta(v_{rem}) \longleftarrow \delta(v_{rem}) - 1$;
      end
    end
  end
  $V_S \longleftarrow V_S \backslash S$;
  if $|V_S| > 0$ **then**
    /* Sélection de la meilleure arête (localement) à introduire dans $\mathcal{T}_k$ */
    $(u_{add}, v_{add}) \longleftarrow (0, 0)$; $inc\_min \longleftarrow 3$;
    for $(u, v) \in E \backslash \mathcal{T}_k$ **do**
      if $u \in S$ `and` $v \notin S$ **then**
        $inc \longleftarrow 0$;
        if $\delta(u) = 2$ **then** $inc \longleftarrow inc + 1$;
        if $\delta(v) = 2$ **then** $inc \longleftarrow inc + 1$;
        if $inc < inc\_min$ **then** $inc\_min \longleftarrow inc$; $(u_{add}, v_{add}) \longleftarrow (u, v)$;
      end
    end
    /* Ajout de l'arête (u_{add}, v_{add}) à $\mathcal{T}_k$ */
    $\mathcal{T}_k \longleftarrow \mathcal{T}_k \cup \{(u_{add}, v_{add})\}$; $\delta(u_{add}) \longleftarrow \delta(u_{add}) + 1$; $\delta(v_{add}) \longleftarrow \delta(v_{add}) + 1$;
  end
end
$f_k \longleftarrow |\{v \in V : \delta(v) \geq 3\}|$;
return $(\mathcal{T}_k, f_k)$;

Algorithme 6: $Repair_{MBV}$: fonction de réparation pour MBV.

4.3.3 Génération de coupes à faible coût computationnel

L'algorithme de plans coupants présenté à la section 4.3.1 est amélioré sur la base des observations suivantes.

1. Le déroulement de l'algorithme de plans coupants peut être vu comme une succession de résolutions de programmes linéaires en variables binaires. A l'exception de la dernière, le résultat de chaque itération est un ensemble de coupes à ajouter à la formulation courante du problème maître, qui sera résolue lors de l'itération suivante. Ainsi, chaque itération est susceptible de requérir du temps et des ressources de calcul importantes.
2. Les graphes creux constituent les instances les plus difficiles à résoudre comme le montrent les résultats de la section 4.5. Mais bien que le nombre de contraintes de connexité (4.6) et/ou de cycle (4.9) soit trop élevé pour qu'on puisse envisager de les ajouter exhaustivement au problème maître, leur nombre diminue considérablement lorsque la densité des graphes diminue.

Ces observations suggèrent qu'il peut être utile de générer des coupes à faible coût computationnel au cours de certaines itérations, c'est-à-dire sans attendre d'obtenir la solution optimale du problème maître. Bien sûr, si les coupes ainsi obtenues se fondent sur une solution très éloignée de la solution optimale du problème maître, il est peu probable que ces coupes soient utiles car aucune bonne solution au problème maître (pour l'itération courante comme pour les itérations à venir) n'est susceptible d'être affectée par les mêmes composantes connexes et les mêmes cycles que la mauvaise solution ayant servi de base à la génération des coupes. Cela soulève la question du compromis à trouver entre le coût computationnel et la qualité des solutions produites lors des itérations de l'algorithme de plans coupants. A cet égard, un algorithme de plans coupants classique privilégie exclusivement la qualité des solutions (puisque la solution optimale est requise), au détriment du coût computationnel, qui peut être très élevé.

Un compromis est trouvé empiriquement en imposant les contraintes suivantes.

- Contrôle du coût computationnel
  Une limite de temps ajustable est imposée au solver pour résoudre le problème maître. Cependant, cette limite de temps n'entre en vigueur qu'à partir du moment où une solution entière a été trouvée. En outre, toutes les α itérations, une rallonge temporelle est accordée au solver afin de permettre l'obtention de solutions de qualité, en s'appuyant sur les coupes générées depuis lors.
- Contrôle de la qualité des solutions produites
  La contrainte suivante est ajoutée au problème maître :

$$\sum_{v \in V} y_v \leq f_Q \tag{4.10}$$

  Où f_Q désigne le nombre maximum de sommets-branche que l'on autorise dans la solution. Ce paramètre est ajusté après chaque itération comme suit :

$$f_Q \longleftarrow \left\lceil \frac{f_{LB} + f_{UB}}{2} \right\rceil$$

  Où f_{LB} est la meilleure borne inférieure disponible sur le nombre de sommets-branche d'une solution optimale pour MBV, et f_{UB} est l'objectif de la meilleure solution faisable disponible.
  La présence de la contrainte (4.10) est susceptible de rendre le problème non faisable si la valeur de f_Q est trop basse. Dans ce cas, f_{LB} est remplacée par $f_Q + 1$, et f_Q est également mise à jour à l'aide de la formule ci-dessus. Les tentatives visant à exiger une qualité plus importante sont apparues contre-productives, dans la mesure où conclure à la non faisabilité d'un problème peut nécessiter un temps de calcul très important, en particulier lorsque f_Q est proche de la valeur optimale de l'objectif.

L'implémentation de ces caractéristiques requiert une structure de contrôle plus complexe, principalement pour ajuster le temps de calcul maximal alloué au solver, tenir compte de la non faisabilité, et aussi parce que l'obtention d'une solution connexe au problème maître n'est désormais plus suffisante pour démontrer son caractère optimal. Par conséquent, les quatre modifications suivantes sont opérées sur l'algorithme 4.

1. La phase d'initialisation (lignes 1 à 3) est remplacée par

> Le problème maître est initialement défini par les équations $(4.1) - (4.5), (4.7) - (4.8)$ et (4.10);
> $stop \longleftarrow$ `false` ;
> $it \longleftarrow 0$;
> $add\_more\_time \longleftarrow 0$;
> $f_{LB} \longleftarrow 0$;
> $f_{UB} \longleftarrow n$;

Où *it* est un compteur d'itérations, et *add_more_time* est un entier permettant d'allouer davantage de temps au solver pour résoudre le problème maître dans le cas où sa solution est un arbre, mais n'est pas optimale (ou bien est optimale sans qu'on n'ait pu le démontrer à l'instant considéré).

2. L'appel au solver (ligne 5) est remplacé par

```
f_Q ⟵ ⌈(f_LB + f_UB)/2⌉;
Mise à jour du second membre de la contrainte Σ_{v∈V} y_v ≤ f_Q;
if it mod α ≠ 0 then
    actual_time ⟵ timelimit;
else
    actual_time ⟵ ⌈timelimit + 60 × it/α⌉;
end
Paramétrer le solver de façon à ce qu'il retourne la meilleure solution entière trouvée au plus tard
actual_time secondes après l'obtention de la première solution entière;
(T, f) ⟵ Solution du problème maître;
if le problème maître n'admet pas de solution then
    f_LB ⟵ f_Q + 1;
else
    it ⟵ it + 1;
    if f > f_LB then
        f_LB ⟵ f;
    end
end
```

On alloue *actual_time* secondes au solver pour lui permettre de déterminer une solution optimale au problème maître (ou à défaut une bonne solution) à compter de l'instant où la première solution entière est trouvée. La valeur « ordinaire » de *actual_time* est *timelimit*, cependant le solver se voit allouer plus de temps toutes les α itérations, afin de tirer profit de toutes les coupes produites depuis le début de la recherche, et plus généralement pour rééquilibrer le compromis entre coût computationnel et qualité des solutions. Le second membre de la contrainte contrôlant la qualité (4.10) est également réajusté si la problème maître est infaisable, sinon la meilleure borne inférieure est mise à jour dès lors que $f > f_{LB}$.

3. Les opérations à effectuer lorsque $\mathcal{T}$ est un arbre (lignes 8 à 10) sont désormais les suivantes

```
if f_LB = f_UB then
    T_UB ⟵ T;
    f_UB ⟵ f;
    stop ⟵ true;
else
    add_more_time ⟵ add_more_time + 1;
end
```

4. L'instruction ci-dessous doit être insérée entre les lignes 12 et 13

```
add_more_time ⟵ 0;
```

En effet, les lignes 12 à 23 de l'algorithme 4 ne sont exécutées que si $\mathcal{T}$ est non connexe, par conséquent à cette étape de l'algorithme il n'est plus nécessaire d'accorder plus de temps au solver, puisque la solution courante n'est plus un arbre. Par conséquent *add_more_time* est remis à zéro.

A partir de maintenant, CPA_{MBV} désigne l'implémentation de l'algorithme de plans coupants auquel les modifications ci-dessus ont été apportées.

4.3.4 Recherche tabou pour l'amélioration des solutions faisables

La fonction de réparation est suivie d'une recherche tabou [33] visant à améliorer la qualité de la borne supérieure intervenant dans l'algorithme de plans coupants. Etant donné que plusieurs solutions faisables sont retournées par la fonction de réparation à chaque itération, il est souhaitable que la recherche tabou utilisée pour tenter d'améliorer

chacune de ces solutions soit rapide. Cette recherche s'appuie sur deux listes tabou et effectue une recherche dans deux directions. La première privilégie l'intensification, et la seconde assure la diversification [63].

Le premier niveau de la recherche commence par la création d'une liste des sommets-branche, séparée en deux parties. Tous les sommets-branche dont le degré est inférieur à un seuil noté D_{max}, sont triés par ordre croissant de leur degré et constituent la première partie de la liste. Les autres sommets-branche la complètent, dans un ordre quelconque. Le premier niveau de la recherche consiste à tenter de réduire le degré des sommets de la première partie de la liste triée, en remplaçant les arêtes incidentes au sommet courant par de nouvelles arêtes assurant la connexité sans créer de nouveaux sommets-branche. Le premier remplacement possible est alors opéré, la liste est mise à jour en conséquence et la recherche reprend à partir du premier sommet-branche de la liste. Si aucun remplacement n'est possible, alors le degré du sommet-branche considéré ne peut pas être réduit et le sommet suivant dans la liste fait l'objet du même traitement. Pour éviter le cyclage, un compteur est associé à chaque sommet-branche de la liste et est initialisé à zéro. Chaque fois qu'une arête est ajoutée, les compteurs associés aux deux extrémités sont incrémentés de un. Si un compteur dépasse le seuil lim_s, alors aucune nouvelle modification impliquant le sommet-branche associé à ce compteur n'est plus tentée, même si ce sommet appartient à la première partie de la liste. Ainsi, tous les sommets-branche de la première partie de la liste dont le compteur dépasse lim_s constituent une liste tabou, qui est appelée « première liste tabou ». Bien sûr, si le nombre de sommets-branche atteint zéro, alors la solution correspondante est optimale et la recherche tabou se termine. Le processus se répète jusqu'à ce qu'il ne soit plus possible de réduire le degré d'un sommet-branche de la première partie de la liste. Ensuite, le second niveau de la recherche commence.

Le second niveau de la recherche tente de transformer un sommet-branche en un sommet de degré inférieur ou égal à deux, même si cela provoque la création de nouveaux sommets-branche. Comme précédemment, cette partie de la recherche commence par le premier sommet-branche de la liste triée, mais elle considère aussi les sommets-branche non triés de la seconde partie de la liste si le degré d'aucun sommet-branche de la première partie de la liste ne peut être rendu inférieur ou égal à deux. Le second niveau de la recherche tente de réduire le degré du sommet-branche considéré en remplaçant ses arêtes incidentes par d'autres arêtes permettant de connecter les deux composantes connexes créées par le retrait de l'arête originale, et dont l'insertion ne provoque pas une augmentation du nombre de sommets-branche supérieure à un, ni ne transforme un sommet-branche de la seconde liste tabou (qui sera décrite plus bas) en un sommet-branche. Le premier remplacement possible est opéré. Si aucun remplacement n'est possible, le sommet-branche suivant fait l'objet de la même recherche. Le processus se répète jusqu'à ce que le degré d'un sommet-branche ait pu être rendu inférieur ou égal à deux, ou que tous les sommets-branche aient été traités sans succès. Dans ce dernier cas, la recherche tabou se termine. Si un sommet-branche a pu être transformé en un sommet dont le degré est inférieur ou égal à deux, alors ce sommet est placé dans la seconde liste tabou. Les sommets de cette liste ne peuvent être à nouveau transformés en sommets-branche. L'ancien sommet-branche considéré fait l'objet de nouvelles tentatives pour ramener son degré à un avec une probabilité p_{d1}, sinon il n'est plus modifié. Ensuite, la première liste tabou est vidée, c'est-à-dire que tous les compteurs associés aux sommets-branche sont remis à zéro, et le premier niveau de la recherche est lancé.

Chaque fois que la recherche tabou construit une solution de meilleure qualité que les solutions précédentes, les deux listes tabou sont vidées afin d'intensifier la recherche dans le voisinage de cette solution. La recherche tabou se termine si la meilleure solution n'est pas améliorée à l'issue de it_{max} itérations de la recherche de premier niveau. Comme indiqué précédemment, la recherche tabou se termine également si une solution sans sommet-branche est trouvée, ou si la recherche de second niveau ne parvient à faire diminuer le degré d'aucun sommet-branche.

L'algorithme de plans coupants intégrant cette recherche tabou est noté CPA^{TS}_{MBV}.

4.4 Méthodes de résolution de MDS

Une formulation par programmation linéaire en nombres entiers pour MDS est disponible dans [14]. Les deux approches proposées pour aborder MDS sont appelées CPA_{MDS} et CPA^{TS}_{MDS}, elles sont très proches de CPA_{MBV} et CPA^{TS}_{MBV} qui viennent d'être présentées pour résoudre MBV. Une nouvelle famille de variables de décision entières est introduite pour représenter le degré des sommets-branche : pour tout $v \in V$, z_v est égal au degré du sommet v si ce sommet est un sommet-branche, sinon z_v est égal à zéro.

MDS peut être formulé par le programme linéaire en nombres entiers noté IP_{MDS} et défini par :

$$\text{Minimiser} \quad \sum_{v \in V} z_v \tag{4.11}$$

$$\begin{aligned}
\text{Sous les contraintes} \quad & \sum_{e \in I(v)} x_e \geq 1 && \forall v \in V \\
& \sum_{e \in E} x_e = n - 1 && \\
& y_v = 0 && \forall v \in V, |I(v)| \leq 2 \\
& \sum_{e \in I(v)} x_e - 2 \leq (|I(v)| - 2)\, y_v && \forall v \in V \\
& \sum_{e \in I(v)} x_e - 2 + 2y_v \leq z_v && \forall v \in V \qquad (4.12) \\
& \sum_{e \in CB(S)} x_e \geq 1 && \forall S \subset V \\
& x_e \in \{0, 1\} && \forall e \in E \\
& y_v \in \{0, 1\} && \forall v \in V \\
& z_v \in \mathbb{N} && \forall v \in V \qquad (4.13)
\end{aligned}$$

(4.11) est la fonction objectif de MDS, les équations non numérotées sont les mêmes que celles de IP_{MBV}. Les contraintes (4.12) et (4.13) assurent que z_v est égal au degré de v si v est un sommet-branche.

Comme dans le cas de MBV, le problème maître pour MDS est construit à partir de IP_{MDS} en ignorant les contraintes de connexité. La génération conjointe de coupes de connexité et de cycle est également adoptée, de sorte que l'algorithme de plans coupants pour MDS est identique à celui proposé dans la section 4.3, à l'exception des quatre points suivants :

1. Dans l'algorithme 4, la ligne 1 est remplacée par

   > Le problème maître est initialement défini par les équations (4.2) − (4.5), (4.7) − (4.8) et (4.11) − (4.14);

2. Dans les algorithmes 4 et 6, f_{UB} désigne maintenant la somme des degrés des sommets-branche de la meilleure solution faisable disponible.

3. La contrainte contrôlant la qualité des solutions produites est maintenant

$$\sum_{v \in V} z_v \leq f_Q \tag{4.14}$$

   Où f_Q désigne la valeur maximale de la somme des degrés des sommets-branche dans toute solution. Ce paramètre est ajusté après chaque itération comme suit :

$$f_Q \longleftarrow \left\lceil \frac{f_{LB} + f_{UB}}{2} \right\rceil$$

   Où f_{LB} est la meilleure borne inférieure disponible sur la somme des degrés des sommets-branche d'une solution optimale, et f_{UB} est l'objectif de la meilleure solution faisable disponible.

4. La fonction de réparation $Repair_{MBV}$ est adaptée à MDS comme suit. On crée une fonction de réparation spécifique à MDS appelée $Repair_{MDS}$ en copiant l'algorithme 6 et en opérant les modifications suivantes.
   - Les lignes 20 et 21 sont remplacées par

   > **if** $\delta(u') = 3$ **then** $dec \longleftarrow dec + 3$;
   > **else if** $\delta(u') > 3$ **then** $dec \longleftarrow dec + 1$;
   > **if** $\delta(v') = 3$ **then** $dec \longleftarrow dec + 3$;
   > **else if** $\delta(v') > 3$ **then** $dec \longleftarrow dec + 1$;

Le retrait d'une arête dont une extrémité est un sommet de degré trois a pour conséquence une diminution de trois de la valeur de la fonction objectif. Ce cas correspond à la transformation d'un sommet-branche en un sommet de degré deux. Si le degré de l'extrémité de l'arête considérée était strictement supérieur à trois avant son retrait, alors la valeur de la fonction objectif ne diminue que de un. La contribution de l'autre extrémité de l'arête suit la même règle.

- La ligne 30 est remplacée par

  $(u_{add}, v_{add}) \longleftarrow (0,0)\,;\ inc\_min \longleftarrow 7;$

  En effet, l'ajout d'une arête peut conduire à une augmentation de 6 de la somme des degrés des sommets-branche, si les deux extrémités de l'arête étaient initialement de degré deux. En initialisant inc_min à sept, on s'assure que toute arête candidate à l'insertion conduira à une augmentation de la valeur de la fonction objectif inférieure à sept.
- Les lignes 34 et 35 sont remplacées par

  if $\delta(u) = 2$ **then** $inc \longleftarrow inc + 3;$
  else if $\delta(u) > 2$ **then** $inc \longleftarrow inc + 1;$
  if $\delta(v) = 2$ **then** $inc \longleftarrow inc + 3;$
  else if $\delta(v) > 2$ **then** $inc \longleftarrow inc + 1;$

  Lors de l'ajout d'une arête, l'augmentation de la fonction objectif dûe à l'une de ses extrémités est de trois si cette extrémité était initialement de degré deux. Ce cas correspond à la création d'un nouveau sommet-branche. Si le degré de ce sommet était initialement supérieur à deux, alors l'augmentation de la valeur de la fonction objectif est de un. La contribution de l'autre extrémité de l'arête suit la même règle.
- La recherche tabou proposée pour MDS est très similaire à celle qui a été présentée pour MBV. Elle se fonde également sur deux niveaux de recherche, deux listes tabou, et a les mêmes critères d'arrêt et de gestion des listes tabou. Le second niveau de la recherche est le même que pour MBV. Le premier niveau présente cependant les différences suivantes. La première liste tabou est désormais entièrement triée. Le premier niveau de recherche considère tous les sommets-branche, et non plus seulement ceux dont le degré est inférieur ou égal à un seuil. Le premier niveau consiste à tenter de réduire le degré des sommets-branche en remplaçant les arêtes incidentes au sommet courant par des arêtes reliant les composantes connexes créées par le retrait des arêtes initiales, mais dont l'insertion ne conduit pas à la création de nouveaux sommets-branche, et qui satisfont l'une des conditions suivantes (par ordre de préférence décroissante) :

  (a) Aucune des extrémités de l'arête insérée ne devient un sommet-branche,

  (b) Si l'autre extrémité de l'arête retirée est également un sommet-branche, alors au plus une extrémité de l'arête insérée peut être un sommet-branche,

  (c) Si le sommet-branche considéré est de degré inférieur ou égal à D_{max}, alors le choix de l'arête à insérer obéit aux mêmes règles que dans le cas de MBV.

  La première arête satisfaisant l'une de ces conditions est insérée afin de remplacer l'arête existante candidate au retrait.

4.5 Résultats numériques

4.5.1 Protocole expérimental

La formulation fondée sur la programmation linéraire en nombres entiers et la meilleure heuristique proposée dans [14] sont comparées aux algorithmes de plans coupants introduits dans la section 4.3.3 avec et sans la recherche tabou pour résoudre MBV dans la section 4.5.2, et MDS dans la section 4.5.3. Tous les algorithmes sont codées en C et exécutés sur un ordinateur utilisant un processeur Intel Pentium IV cadencé à 3.2 GHz muni de 1 gigaoctet de RAM sous Microsoft Windows XP, le solver est XPRESS-MP utilisé au travers de la librairie XPRESS-BCL [72].

Les instances utilisées dans cette section ont été produites comme dans [14], à l'aide des générateurs *Genmax*, *Netgen* et *Random*, mis en ligne par DIMACS (`http://dimacs.rutgers.edu/`). *Genmax* et *Netgen* sont des générateurs d'instances de problèmes de flot produisant des graphes dont les arcs sont générées aléatoirement (leur capacité suit une loi de distribution normale). La direction des arcs a été ignorée, ainsi que leur capacité. Pour chaque générateur, les huit valeurs suivantes de $|V|$ ont été considérées : $\{20, 30, 40, 50, 100, 300, 500, 1000\}$. Pour chaque valeur de $|V|$, les cinq densités d suivantes ont été exploitées : $\{1.5, 2, 4, 10, 15\}$, où la densité est égale au rapport du nombre d'arêtes sur le nombre de sommets du graphe. En raison de restrictions internes, le générateur *Random*

s'est avéré incapable de produire des instances pour $\{|V| = 20, d = 10\}$, $|\{V| = 20, d = 15\}$ et $\{|V| = 30, d = 15\}$. Ainsi, quarante couples $(|V|, d)$ ont été considérés pour chaque générateur, à l'exception de *Random* qui n'en compte que trente-sept. Pour chaque combinaison, cinq instances sont produites, ce qui conduit à un total de cinq cent quatre vingt cinq instances. Les résultats obtenus sont reportés à travers une moyenne calculée sur les cinq instances associées au même couple $(|V|, d)$. Le nom de ces ensembles d'instances est de la forme `typ_XXXX_YYYYY`, où `typ` désigne le type de générateur, et vaut `gen` pour *Genmax*, `net` pour *Netgen* et `ran` pour *Random*. `XXXX` est le nombre de sommets du graphe et `YYYYY` est le nombre d'arêtes.

La génération de coupes à faible coût computationnel est régie par deux paramètres, α and *timelimit*. Ils sont fixés comme suit : $\alpha = 5$, et *timelimit* est fixé à $\lceil \frac{n}{25} \rceil$ si le nombre d'arêtes est inférieur à $2.2n$ avec $n \geq 300$, sinon *timelimit* est égal à $+\infty$ (c'est-à-dire que la solution optimale du problème maître est requise). Comme le montrent les résultats, les graphes creux sur plus de 300 sommets constituent les instances les plus difficiles à résoudre. Le temps alloué au solver pour résoudre le problème maître est plus court pour ces instances, afin de permettre la génération d'un plus grand nombre de coupes. Cependant, afin de trouver un juste équilibre entre la qualité des coupes produites et le temps de calcul, *timelimit* est proportionnel au nombre de sommets du graphe. Ainsi, la valeur numérique de α et la formule régissant *timelimit* dépendent de l'ordinateur et des performances du solver, comme c'est souvent le cas lors de l'implémentation d'une métaheuristique. Ces choix ont été faits de manière empirique, et restent les mêmes quels que soit le type d'instance.

Pour ce qui concerne les paramètres propres à la recherche tabou, les valeurs choisies sont $D_{max} = 5$, $lim_s = 4$, $p_{d1} = 0.75$, $it_{max} = max(2n, 200)$. La taille de la liste tabou est de $max(3N_{br}, 50)$ itérations pour le premier niveau de recherche, où N_{br} désigne le nombre de sommets-branche de la solution courante. Là encore, ces paramètres ont été fixés de manière empirique.

On choisit de limiter le temps de calcul dédié à la résolution de chaque instance à une heure. Si aucune solution optimale n'a été trouvé lorsque ce délai est écoulé, on retourne la meilleure solution trouvée et la meilleure borne inférieure disponible.

4.5.2 Résultats obtenus pour MBV

Les tables 4.3, 4.4 et 4.5 présentent les résultats obtenus pour MBV. La première colonne de ces tables est le nom de l'instance, les instances apparaissant par nombre de sommets et d'arêtes croissant. Les trois colonnes suivantes mesurent la qualité des solutions retournées en indiquant l'écart maximum (en pourcentage) entre la meilleure solution retournée et la meilleure borne inférieure calculée. Cet écart est noté ΔIP_{MBV} pour IP_{MBV} (le programme linéaire en nombres entiers proposé dans [14]), ΔCPA_{MBV} pour l'algorithme de plans coupants muni de la fonction de réparation, et ΔCPA^{TS}_{MBV} pour la version utilisant la recherche tabou. Le nombre de solutions optimales trouvées est indiqué entre parenthèses. Par exemple, 0.00 (5) signifie que l'approche concernée a retourné une solution optimale pour les cinq instances de type `typ` ayant `XXXX` sommets et `YYYYY` arêtes. La colonne intitulée ΔCA mesure l'amélioration de la qualité des solutions produites par CPA^{TS}_{MBV} (en pourcentage) par comparaison avec la *Combined Approach* (CA), qui est la meilleure heuristique proposée dans [14] pour MBV. L'amélioration atteint 100% lorsque CA retourne une solution comportant un ou plusieurs sommets-branche alors qu'une chaîne hamiltonienne a été identifiée par CPA^{TS}_{MBV}. Les trois colonnes suivantes indiquent le temps de calcul moyen pour une instance, exprimé en secondes, sur les instances dont la solution optimale a été trouvée. Si aucune solution optimale n'a été trouvée pour les cinq instances dans la limite de temps impartie, « − » est affiché dans la table. La dernière colonne donne le temps de calcul moyen de l'heuristique *Combined Approach* sur les cinq instances.

La dernière ligne de ces tables résume les résultats sur les instances ayant plus de 300 sommets comme suit : les colonnes 2 à 4 indiquent le nombre de solutions optimales trouvées, en pourcentage. Les colonnes 6 à 8 donnent le temps de calcul moyen pour déterminer une solution optimale (ce calcul ne tient pas compte des instances pour lesquelles la solution optimale n'a pas été trouvée), et la dernière colonne est le temps de calcul moyen de CA.

Les tables 4.3, 4.4 et 4.5 montrent que toutes les approches exactes sont capables de trouver une solution optimale pour les instances telles que $|V| \leq 100$. C'est pour cela que seules les instances sur plus de 300 sommets sont prises en compte pour fournir des comparaisons globales répertoriées dans la dernière ligne de ces tables. Néanmoins les résultats sur les petites instances (quel que soit leur type) permettent d'observer que les approches basées sur les algorithmes de plans coupants sont parfois jusqu'à cent fois plus rapides que IP_{MBV}. La recherche tabou semble ne pas apporter de gains significatifs sur ces instances de petite taille.

Pour les instances sur 300 sommets, la supériorité des algorithmes de plans coupants sur IP_{MBV} se confirme, en particulier sur les graphes denses. Cela tient au fait que ces graphes sont bien souvent hamiltoniens, et que la découverte d'une chaîne hamiltonienne par la fonction de réparation (ou par la recherche tabou) met un terme à la recherche. IP_{MBV} est en revanche incapable de tirer parti de cette caractéristique, et voit son temps de calcul augmenter fortement avec la taille de l'instance. Cette tendance est particulièrement visible sur les graphes de

grande taille avec $d \geq 10$, où IP_{MBV} ne trouve parfois aucune solution faisable en moins d'une heure alors que CPA^{TS}_{MBV} détermine une solution optimale (une chaîne hamiltonienne) en moins de 6 secondes. L'accélération de la convergence conférée par la recherche tabou est manifeste sur les instances de grande taille : la réduction du temps de calcul par rapport à CPA_{MBV} peut atteindre 80% sur `gen_1000_10000`. Les résultats sur les instances produites par *Netgen* montrent tout l'intérêt des approches basées sur les plans coupants car une solution optimale est obtenue pour toutes les instances. Même si CA se distingue par des temps de calcul très modestes, la qualité des solutions produites est particulièrement mauvaise même sur les instances très denses où une chaîne hamiltonienne est obtenue assez rapidement par les algorithmes basés sur les plans coupants.

Ces résultats confirment que les instances les plus difficiles pour MBV sont celles dont la densité est basse, le nombre de sommets élevé, et qui ont été produites par *Random*. Mais même lorsqu'aucune approche exacte ne parvient à trouver de solution optimale, les algorithmes basés sur les plans coupants retournent des solutions de qualité bien supérieure à celles retournées par IP_{MBV} (en terme d'écart maximum à la solution optimale). CPA^{TS}_{MBV} s'impose comme l'approche la plus efficace, la recherche tabou permettant de réduire l'écart d'un facteur 3. Bien que le nombre de solutions optimales trouvées pour les instances produites avec *Genmax* et *Random* soit très proche avec et sans la recherche tabou, l'amplitude de l'écart maximal à la solution optimale est généralement deux fois plus faible lorsque la recherche tabou est utilisée.

La comparaison en termes de temps de calcul est également à l'avantage de CPA^{TS}_{MBV}, qui peut être 6 fois plus rapide que CPA_{MBV} à nombre égal de solutions optimales trouvées dans le cas des instances *Random*. Le temps de calcul moyen d'une solution optimale par CPA^{TS}_{MBV} est même 36 fois plus faible que pour IP_{MBV}, ce qui lui permet d'obtenir une solution optimale pour presque deux fois plus d'instances.

4.5.3 Résultats obtenus pour MDS

Les tables 4.6, 4.7 et 4.8 présentent les résultats obtenus pour MDS. Les conclusions que l'on peut en tirer sont assez similaires à celles qui ont été établies pour MBV, à l'exception de ce qui suit.

La meilleure heuristique proposée dans [14] est l'algorithme *Edge Weight* (noté dans les tables EW), qui remplace CA. Le nombre d'instances de plus de 300 sommets dont on parvient à trouver une solution optimale est globalement plus élevé pour MDS, ce qui suggère que MDS est plus facile que MBV sur les instances produites par *Genmax* et *Netgen*. De plus, l'avantage apporté par la recherche tabou, bien que toujours notable, est un peu moins décisif qu'il ne l'était pour MBV. Il convient cependant d'être prudent dans l'interprétation de la dernière ligne de la table 4.8 concernant CPA_{MDS} et CPA^{TS}_{MDS}. En effet, il apparaît que le temps de calcul moyen d'une solution optimale pour les instances produites par *Random* est plus faible pour CPA_{MDS} que pour CPA^{TS}_{MDS}, mais cela est dû au fait que CPA^{TS}_{MDS} a retourné plus de solutions optimales que CPA_{MDS}, et que ces solutions optimales supplémentaires, difficiles à obtenir, ont nécessité un temps de calcul important, faisant ainsi augmenter la moyenne des temps de calcul. On voit qu'une telle situation se présente pour l'une des cinq instances `ran_0500_01000`.

En définitive, on observe que les méthodes à base de plans coupants proposées dans ce chapitre conduisent à des solutions de qualité nettement supérieure à celles retournées par les approches présentées dans [14].

4.6 Conclusion

Ce chapitre aborde deux problèmes proches (mais indépendants) visant à minimiser le coût des appareils opto-électroniques à installer pour mettre en œuvre le multiplexage en longueur d'onde dans un réseau de fibre optique destiné à répondre aux besoins de la communication multicast, sous deux hypothèses différentes concernant le coût des équipements en question. Les approches proposées semblent très efficaces en termes de qualité des solutions retournées comme en termes de temps de calcul. On peut avancer trois raisons à cela. La première tient à la génération de coupes à coût computationnel faible, qui vise à éviter que la recherche ne stagne lorsque la résolution optimale du problème maître devient difficile, tout en produisant des coupes qui s'avèrent utiles pour la suite de la recherche. Cette caractéristique contribue également à améliorer la borne inférieure. La seconde raison de l'efficacité des approches proposées tient à la fonction de réparation qui exploite les solutions (non faisables) produites par le problème maître à chaque itération pour construire des solutions faisables dont les caractéristiques sont proches de celles retournées par le problème maître. La qualité de ces solutions est améliorée par une recherche tabou, ce qui contribue à améliorer la borne supérieure. Enfin, la troisième raison tient à la coopération entre ces deux processus : la contrainte assurant le contrôle de la qualité des solutions retournées par le problème maître exploite l'objectif des solutions retournées par la recherche tabou, alors que la fonction de réparation tire directement profit des solutions produites par l'algorithme de plans coupants. Cette collaboration mutuellement profitable à un algorithme exact et à une métaheuristique fait apparaître le potentiel considérable des *matheuristiques* pour la résolution des problèmes

d'optimisation combinatoire, en dépassant les schémas classiques sans collaboration reposant soit sur une méthode exacte faisant office d'heuristique en retournant une borne inférieure et la meilleure solution trouvée après une durée donnée, soit sur l'utilisation d'une borne inférieure et d'une métaheuristique. A cet égard, la comparaison avec les résultats de [14] où un programme linéaire en nombres entiers et des heuristiques sont proposés indépendamment est clairement en faveur des approches matheuristiques développées dans ce chapitre. De telles approches peuvent être employées pour aborder d'autres variantes difficiles du problème de l'arbre couvrant, comme le problème de l'arbre couvrant de poids minimum avec des contraintes sur le degré des sommets, ou encore le problème de l'arbre dominant de poids minimum.

TABLE 4.3 – Résultats pour MBV sur les instances produites par *Genmax*

Instance	Ecart LB/UB en pourcents				Temps de calcul moyen (sec.)			
	ΔIP_{MBV}	ΔCPA_{MBV}	ΔCPA^{TS}_{MBV}	ΔCA	IP_{MBV}	CPA_{MBV}	CPA^{TS}_{MBV}	CA
gen_0020_00030	0.00 (5)	0.00 (5)	0.00 (5)	48.33	0.09	0.56	0.21	0.00
gen_0020_00040	0.00 (5)	0.00 (5)	0.00 (5)	73.33	0.10	0.23	0.13	0.00
gen_0020_00080	0.00 (5)	0.00 (5)	0.00 (5)	80.00	0.10	0.20	0.12	0.00
gen_0020_00200	0.00 (5)	0.00 (5)	0.00 (5)	0.00	0.13	0.17	0.14	0.00
gen_0020_00300	0.00 (5)	0.00 (5)	0.00 (5)	0.00	0.27	0.18	0.14	0.00
gen_0030_00045	0.00 (5)	0.00 (5)	0.00 (5)	51.76	0.18	0.23	0.18	0.00
gen_0030_00060	0.00 (5)	0.00 (5)	0.00 (5)	84.33	0.40	0.20	0.13	0.00
gen_0030_00120	0.00 (5)	0.00 (5)	0.00 (5)	89.33	0.41	0.19	0.13	0.00
gen_0030_00300	0.00 (5)	0.00 (5)	0.00 (5)	40.00	0.83	0.15	0.14	0.00
gen_0030_00450	0.00 (5)	0.00 (5)	0.00 (5)	0.00	0.51	0.16	0.14	0.00
gen_0040_00060	0.00 (5)	0.00 (5)	0.00 (5)	61.71	0.31	0.28	0.18	0.00
gen_0040_00080	0.00 (5)	0.00 (5)	0.00 (5)	87.67	0.47	0.17	0.13	0.00
gen_0040_00160	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.59	0.15	0.14	0.00
gen_0040_00400	0.00 (5)	0.00 (5)	0.00 (5)	60.00	0.88	0.16	0.14	0.00
gen_0040_00600	0.00 (5)	0.00 (5)	0.00 (5)	60.00	1.60	0.17	0.14	0.00
gen_0050_00075	0.00 (5)	0.00 (5)	0.00 (5)	58.92	0.70	0.51	0.34	0.00
gen_0050_00100	0.00 (5)	0.00 (5)	0.00 (5)	86.31	0.71	0.31	0.14	0.00
gen_0050_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.83	0.16	0.14	0.00
gen_0050_00500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.51	0.23	0.15	0.00
gen_0050_00750	0.00 (5)	0.00 (5)	0.00 (5)	40.00	1.68	0.17	0.15	0.00
gen_0100_00150	0.00 (5)	0.00 (5)	0.00 (5)	61.70	12.11	5.54	2.53	0.00
gen_0100_00200	0.00 (5)	0.00 (5)	0.00 (5)	84.39	11.22	6.32	0.45	0.00
gen_0100_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	4.78	0.21	0.15	0.00
gen_0100_01000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	14.42	0.17	0.18	0.00
gen_0100_01500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	13.66	0.20	0.18	0.01
gen_0300_00450	32.84 (0)	1.90 (4)	1.00 (4)	59.35	–	1720.69	1023.95	0.00
gen_0300_00600	63.03 (0)	2.86 (4)	0.00 (5)	85.01	–	1783.08	215.92	0.01
gen_0300_01200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	92.09	12.82	1.06	0.03
gen_0300_03000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	554.34	0.40	0.37	0.11
gen_0300_04500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1345.46	0.87	0.48	0.13
gen_0500_00750	42.39 (0)	8.05 (0)	5.19 (0)	60.07	–	–	–	0.02
gen_0500_01000	71.86 (0)	20.14 (0)	8.94 (1)	85.30	–	–	1810.50	0.06
gen_0500_02000	10.00 (4)	0.00 (5)	0.00 (5)	99.20	548.22	7.42	2.06	0.08
gen_0500_05000	40.00 (3)	0.00 (5)	0.00 (5)	100.00	1809.49	2.12	1.37	0.24
gen_0500_07500	60.00 (2)	0.00 (5)	0.00 (5)	100.00	1375.68	1.12	0.87	0.37
gen_1000_01500	68.94 (0)	18.21 (0)	14.69 (0)	55.24	–	–	–	0.13
gen_1000_02000	80.98 (0)	31.32 (0)	10.56 (0)	83.50	–	–	–	0.18
gen_1000_04000	76.98 (1)	0.00 (5)	0.00 (5)	99.01	2558.30	274.66	4.84	0.40
gen_1000_10000	100.00 (0)	0.00 (5)	0.00 (5)	100.00	–	13.41	2.88	1.02
gen_1000_15000	100.00 (0)	0.00 (5)	0.00 (5)	100.00	–	14.84	5.95	1.66
Sol. opt.	**33.33%**	**70.67%**	**73.33%**		**915.62**	**295.35**	**128.82**	**0.30**

TABLE 4.4 – Résultats pour MBV sur les instances produites par *Netgen*.

Instance	Ecart LB/UB en pourcents				Temps de calcul moyen (sec.)			
	ΔIP_{MBV}	ΔCPA_{MBV}	ΔCPA^{TS}_{MBV}	ΔCA	IP_{MBV}	CPA_{MBV}	CPA^{TS}_{MBV}	CA
net_0020_00030	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.05	0.12	0.12	0.00
net_0020_00040	0.00 (5)	0.00 (5)	0.00 (5)	80.00	0.11	0.12	0.12	0.00
net_0020_00080	0.00 (5)	0.00 (5)	0.00 (5)	20.00	0.16	0.11	0.12	0.00
net_0020_00200	0.00 (5)	0.00 (5)	0.00 (5)	20.00	0.20	0.12	0.12	0.00
net_0020_00300	0.00 (5)	0.00 (5)	0.00 (5)	20.00	0.32	0.12	0.12	0.00
net_0030_00045	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.09	0.12	0.12	0.00
net_0030_00060	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.17	0.13	0.12	0.00
net_0030_00120	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.42	0.13	0.12	0.00
net_0030_00300	0.00 (5)	0.00 (5)	0.00 (5)	80.00	0.67	0.13	0.13	0.00
net_0030_00450	0.00 (5)	0.00 (5)	0.00 (5)	60.00	0.70	0.14	0.13	0.00
net_0040_00060	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.20	0.14	0.12	0.00
net_0040_00080	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.40	0.12	0.12	0.00
net_0040_00160	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.68	0.12	0.12	0.00
net_0040_00400	0.00 (5)	0.00 (5)	0.00 (5)	80.00	1.03	0.13	0.13	0.00
net_0040_00600	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.22	0.14	0.13	0.00
net_0050_00075	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.19	0.13	0.12	0.00
net_0050_00100	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.56	0.13	0.12	0.00
net_0050_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.09	0.12	0.13	0.00
net_0050_00500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.47	0.13	0.15	0.00
net_0050_00750	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.24	0.15	0.15	0.00
net_0100_00150	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.09	0.18	0.17	0.00
net_0100_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	2.75	0.18	0.13	0.00
net_0100_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	4.19	0.15	0.15	0.00
net_0100_01000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	8.95	0.20	0.19	0.00
net_0100_01500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	11.53	0.18	0.18	0.00
net_0300_00450	0.00 (5)	0.00 (5)	0.00 (5)	100.00	8.24	0.38	0.25	0.00
net_0300_00600	0.00 (5)	0.00 (5)	0.00 (5)	100.00	21.96	0.58	0.27	0.00
net_0300_01200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	104.53	1.56	0.73	0.03
net_0300_03000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	504.14	0.33	0.33	0.10
net_0300_04500	20.00 (4)	0.00 (5)	0.00 (5)	100.00	1188.95	0.61	0.47	0.14
net_0500_00750	0.00 (5)	0.00 (5)	0.00 (5)	100.00	22.10	0.78	0.58	0.03
net_0500_01000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	100.59	3.83	0.57	0.06
net_0500_02000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	646.94	3.23	1.85	0.11
net_0500_05000	20.00 (4)	0.00 (5)	0.00 (5)	100.00	1721.58	4.43	1.00	0.23
net_0500_07500	60.00 (2)	0.00 (5)	0.00 (5)	100.00	3001.29	2.64	1.30	0.38
net_1000_01500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	121.11	6.62	8.11	0.09
net_1000_02000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	934.20	12.53	4.92	0.17
net_1000_04000	80.00 (1)	0.00 (5)	0.00 (5)	100.00	2385.70	8.53	4.86	0.38
net_1000_10000	100.00 (0)	0.00 (5)	0.00 (5)	100.00	–	13.76	3.03	1.06
net_1000_15000	100.00 (0)	0.00 (5)	0.00 (5)	100.00	–	7.80	5.47	1.56
Sol. opt.	**74.67%**	**100.00%**	**100.00%**		**577.67**	**4.51**	**2.25**	**0.29**

TABLE 4.5 – Résultats pour MBV sur les instances produites par *Random*.

Instance	Ecart LB/UB en pourcents				Temps de calcul moyen (sec.)			
	ΔIP_{MBV}	ΔCPA_{MBV}	ΔCPA^{TS}_{MBV}	ΔCA	IP_{MBV}	CPA_{MBV}	CPA^{TS}_{MBV}	CA
ran_0020_00030	0.00 (5)	0.00 (5)	0.00 (5)	56.67	0.12	0.15	0.13	0.00
ran_0020_00040	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.13	0.11	0.11	0.00
ran_0020_00080	0.00 (5)	0.00 (5)	0.00 (5)	60.00	0.16	0.11	0.11	0.00
ran_0030_00045	0.00 (5)	0.00 (5)	0.00 (5)	47.00	0.33	0.26	0.27	0.00
ran_0030_00060	0.00 (5)	0.00 (5)	0.00 (5)	95.00	0.13	0.12	0.11	0.00
ran_0030_00120	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.48	0.12	0.12	0.00
ran_0030_00300	0.00 (5)	0.00 (5)	0.00 (5)	20.00	1.05	0.13	0.13	0.00
ran_0040_00060	0.00 (5)	0.00 (5)	0.00 (5)	47.56	0.45	0.28	0.28	0.00
ran_0040_00080	0.00 (5)	0.00 (5)	0.00 (5)	69.67	0.76	0.43	0.17	0.00
ran_0040_00160	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.70	0.12	0.13	0.00
ran_0040_00400	0.00 (5)	0.00 (5)	0.00 (5)	40.00	0.99	0.13	0.13	0.00
ran_0040_00600	0.00 (5)	0.00 (5)	0.00 (5)	0.00	2.58	0.15	0.15	0.00
ran_0050_00075	0.00 (5)	0.00 (5)	0.00 (5)	60.16	0.60	0.39	0.24	0.00
ran_0050_00100	0.00 (5)	0.00 (5)	0.00 (5)	78.00	1.26	0.28	0.12	0.00
ran_0050_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.90	0.12	0.12	0.00
ran_0050_00500	0.00 (5)	0.00 (5)	0.00 (5)	80.00	2.27	0.14	0.14	0.00
ran_0050_00750	0.00 (5)	0.00 (5)	0.00 (5)	40.00	3.11	0.14	0.14	0.00
ran_0100_00150	0.00 (5)	0.00 (5)	0.00 (5)	59.15	11.41	8.10	1.94	0.00
ran_0100_00200	0.00 (5)	0.00 (5)	0.00 (5)	81.52	69.32	5.91	2.01	0.00
ran_0100_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	7.03	0.17	0.14	0.00
ran_0100_01000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	13.60	0.17	0.16	0.00
ran_0100_01500	0.00 (5)	0.00 (5)	0.00 (5)	80.00	25.65	0.17	0.17	0.01
ran_0300_00450	27.39 (0)	1.18 (4)	2.05 (3)	60.84	–	1465.08	435.72	0.00
ran_0300_00600	45.93 (1)	2.86 (4)	0.00 (5)	85.36	2544.13	1269.44	79.70	0.01
ran_0300_01200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	99.72	1.96	1.02	0.03
ran_0300_03000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1104.37	0.46	0.37	0.08
ran_0300_04500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1307.45	0.53	0.46	0.14
ran_0500_00750	44.08 (0)	10.16 (0)	6.15 (0)	59.63	–	–	–	0.02
ran_0500_01000	71.13 (0)	20.94 (0)	10.48 (0)	83.37	–	–	–	0.03
ran_0500_02000	10.00 (4)	0.00 (5)	0.00 (5)	99.31	1032.66	22.94	1.35	0.11
ran_0500_05000	60.00 (2)	0.00 (5)	0.00 (5)	100.00	2613.19	1.23	0.95	0.27
ran_0500_07500	80.00 (1)	0.00 (5)	0.00 (5)	100.00	12.41	1.87	1.89	0.41
ran_1000_01500	66.92 (0)	18.12 (0)	15.36 (0)	53.52	–	–	–	0.11
ran_1000_02000	83.53 (0)	33.12 (0)	11.48 (0)	84.71	–	–	–	0.17
ran_1000_04000	59.70 (2)	0.00 (5)	0.00 (5)	99.64	3184.16	25.34	4.08	0.37
ran_1000_10000	100.00 (0)	0.00 (5)	0.00 (5)	100.00	–	9.82	3.50	1.24
ran_1000_15000	100.00 (0)	0.00 (5)	0.00 (5)	100.00	–	11.16	3.20	2.09
Sol. opt.	**33.33%**	**70.67%**	**70.67%**		**1233.58**	**213.48**	**33.77**	**0.34**

TABLE 4.6 – Résultats pour MDS sur les instances produites par *Genmax*.

Instance	Ecart LB/UB en pourcents				Temps de calcul moyen (sec.)			
	ΔIP_{MDS}	ΔCPA_{MDS}	ΔCPA^{TS}_{MDS}	ΔEW	IP_{MDS}	CPA_{MDS}	CPA^{TS}_{MDS}	EW
gen_0020_00030	0.00 (5)	0.00 (5)	0.00 (5)	44.89	0.06	0.14	0.13	0.00
gen_0020_00040	0.00 (5)	0.00 (5)	0.00 (5)	61.11	0.20	0.16	0.15	0.00
gen_0020_00080	0.00 (5)	0.00 (5)	0.00 (5)	80.00	0.16	0.13	0.12	0.00
gen_0020_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.21	0.14	0.13	0.00
gen_0020_00300	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.32	0.13	0.12	0.00
gen_0030_00045	0.00 (5)	0.00 (5)	0.00 (5)	42.37	0.27	0.23	0.22	0.00
gen_0030_00060	0.00 (5)	0.00 (5)	0.00 (5)	73.33	0.35	0.36	0.23	0.00
gen_0030_00120	0.00 (5)	0.00 (5)	0.00 (5)	80.00	0.23	0.37	0.23	0.00
gen_0030_00300	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.64	0.14	0.13	0.00
gen_0030_00450	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.65	0.14	0.14	0.00
gen_0040_00060	0.00 (5)	0.00 (5)	0.00 (5)	55.71	0.25	0.47	0.28	0.00
gen_0040_00080	0.00 (5)	0.00 (5)	0.00 (5)	81.67	25.73	0.17	0.13	0.00
gen_0040_00160	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.56	0.15	0.14	0.00
gen_0040_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.05	0.17	0.16	0.00
gen_0040_00600	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.68	0.16	0.15	0.00
gen_0050_00075	0.00 (5)	0.00 (5)	0.00 (5)	43.68	0.87	0.96	0.67	0.00
gen_0050_00100	0.00 (5)	0.00 (5)	0.00 (5)	74.44	19.17	0.37	0.16	0.00
gen_0050_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.88	0.14	0.14	0.00
gen_0050_00500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.77	0.17	0.16	0.00
gen_0050_00750	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.79	0.17	0.17	0.00
gen_0100_00150	0.00 (5)	0.00 (5)	0.00 (5)	47.63	27.25	3.82	3.38	0.00
gen_0100_00200	1.33 (4)	0.00 (5)	0.00 (5)	63.08	12.34	2.25	0.52	0.00
gen_0100_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	3.33	0.21	0.17	0.00
gen_0100_01000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	9.05	0.22	0.22	0.01
gen_0100_01500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	6.60	0.19	0.18	0.03
gen_0300_00450	16.03 (0)	0.00 (5)	0.00 (5)	44.81	–	1653.98	1242.78	0.01
gen_0300_00600	34.73 (0)	3.70 (3)	0.93 (4)	64.50	–	470.74	307.90	0.01
gen_0300_01200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	23.91	5.32	0.69	0.05
gen_0300_03000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	110.05	1.22	0.72	0.15
gen_0300_04500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	276.30	2.25	2.07	0.21
gen_0500_00750	98.46 (0)	1.69 (0)	1.16 (0)	45.55	–	–	–	0.03
gen_0500_01000	81.05 (0)	8.15 (0)	2.20 (1)	63.77	–	–	2144.59	0.06
gen_0500_02000	4.00 (4)	0.00 (5)	0.00 (5)	98.92	228.84	41.89	3.33	0.14
gen_0500_05000	40.00 (3)	0.00 (5)	0.00 (5)	100.00	194.66	1.38	1.35	0.39
gen_0500_07500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1198.72	3.86	2.82	0.65
gen_1000_01500	96.79 (0)	5.47 (0)	4.71 (0)	43.76	–	–	–	0.15
gen_1000_02000	98.72 (0)	10.06 (0)	5.23 (0)	61.80	–	–	–	0.22
gen_1000_04000	89.16 (0)	0.00 (5)	0.00 (5)	98.54	–	336.88	70.46	0.49
gen_1000_10000	40.00 (3)	0.00 (5)	0.00 (5)	100.00	952.77	19.27	4.43	1.78
gen_1000_15000	100.00 (0)	0.00 (5)	0.00 (5)	100.00	–	15.50	5.59	2.92
Sol. opt.	**40.00%**	**70.67%**	**73.33%**		**413.42**	**223.02**	**182.68**	**0.49**

TABLE 4.7 – Résultats pour MDS sur les instances produites par *Netgen*.

Instance	Ecart LB/UB en pourcents				Temps de calcul moyen (sec.)			
	ΔIP_{MDS}	ΔCPA_{MDS}	ΔCPA^{TS}_{MDS}	ΔEW	IP_{MDS}	CPA_{MDS}	CPA^{TS}_{MDS}	EW
net_0020_00030	0.00 (5)	0.00 (5)	0.00 (5)	60.00	0.02	0.13	0.12	0.00
net_0020_00040	0.00 (5)	0.00 (5)	0.00 (5)	80.00	0.09	0.12	0.13	0.00
net_0020_00080	0.00 (5)	0.00 (5)	0.00 (5)	60.00	0.06	0.12	0.12	0.00
net_0020_00200	0.00 (5)	0.00 (5)	0.00 (5)	80.00	0.29	0.12	0.12	0.00
net_0020_00300	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.25	0.13	0.13	0.00
net_0030_00045	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.10	0.13	0.12	0.00
net_0030_00060	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.20	0.12	0.12	0.00
net_0030_00120	0.00 (5)	0.00 (5)	0.00 (5)	80.00	0.28	0.13	0.13	0.00
net_0030_00300	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.75	0.14	0.14	0.00
net_0030_00450	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.39	0.14	0.15	0.00
net_0040_00060	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.12	0.13	0.13	0.00
net_0040_00080	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.42	0.13	0.11	0.00
net_0040_00160	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.44	0.13	0.12	0.00
net_0040_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.07	0.15	0.15	0.00
net_0040_00600	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.78	0.15	0.15	0.00
net_0050_00075	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.15	0.13	0.12	0.00
net_0050_00100	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.46	0.12	0.12	0.00
net_0050_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.89	0.15	0.14	0.00
net_0050_00500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	2.22	0.16	0.16	0.00
net_0050_00750	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.13	0.16	0.15	0.00
net_0100_00150	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.85	0.16	0.13	0.00
net_0100_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.79	0.21	0.13	0.00
net_0100_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	3.37	0.22	0.16	0.00
net_0100_01000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	9.00	0.17	0.17	0.00
net_0100_01500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	22.35	0.22	0.21	0.01
net_0300_00450	0.00 (5)	0.00 (5)	0.00 (5)	100.00	9.35	0.81	0.40	0.00
net_0300_00600	0.00 (5)	0.00 (5)	0.00 (5)	100.00	39.55	0.57	0.25	0.00
net_0300_01200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	73.84	1.79	1.09	0.03
net_0300_03000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	47.79	1.16	1.12	0.15
net_0300_04500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	187.73	0.89	0.89	0.20
net_0500_00750	0.00 (5)	0.00 (5)	0.00 (5)	100.00	18.55	2.08	2.13	0.03
net_0500_01000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	252.68	3.85	0.82	0.06
net_0500_02000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	210.41	7.67	2.50	0.14
net_0500_05000	40.00 (3)	0.00 (5)	0.00 (5)	100.00	154.18	4.83	4.01	0.39
net_0500_07500	20.00 (4)	0.00 (5)	0.00 (5)	100.00	196.77	3.50	2.48	0.56
net_1000_01500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	233.06	11.76	12.42	0.14
net_1000_02000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	2021.92	17.52	7.81	0.22
net_1000_04000	60.00 (2)	0.00 (5)	0.00 (5)	100.00	1478.23	19.30	3.76	0.52
net_1000_10000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	905.48	34.36	8.16	1.72
net_1000_15000	80.00 (1)	0.00 (5)	0.00 (5)	100.00	974.10	14.38	10.74	2.82
Sol. opt.	**86.67%**	**100.00%**	**100.00%**		**387.42**	**8.30**	**3.91**	**0.47**

TABLE 4.8 – Résultats pour MDS sur les instances produites par *Random*.

Instance	Ecart LB/UB en pourcents				Temps de calcul moyen (sec.)			
	ΔIP_{MDS}	ΔCPA_{MDS}	ΔCPA^{TS}_{MDS}	ΔEW	IP_{MDS}	CPA_{MDS}	CPA^{TS}_{MDS}	EW
ran_0020_00030	0.00 (5)	0.00 (5)	0.00 (5)	50.00	0.10	0.15	0.13	0.00
ran_0020_00040	0.00 (5)	0.00 (5)	0.00 (5)	60.00	0.07	0.13	0.12	0.00
ran_0020_00080	0.00 (5)	0.00 (5)	0.00 (5)	60.00	0.06	0.12	0.12	0.00
ran_0030_00045	0.00 (5)	0.00 (5)	0.00 (5)	36.31	0.32	0.32	0.30	0.00
ran_0030_00060	0.00 (5)	0.00 (5)	0.00 (5)	95.00	0.54	0.12	0.12	0.00
ran_0030_00120	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.22	0.12	0.12	0.00
ran_0030_00300	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.93	0.13	0.13	0.00
ran_0040_00060	0.00 (5)	0.00 (5)	0.00 (5)	45.53	0.68	0.33	0.34	0.00
ran_0040_00080	0.00 (5)	0.00 (5)	0.00 (5)	66.83	36.44	0.31	0.32	0.00
ran_0040_00160	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.60	0.13	0.13	0.00
ran_0040_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	1.00	0.13	0.14	0.00
ran_0040_00600	0.00 (5)	0.00 (5)	0.00 (5)	100.00	2.73	0.15	0.15	0.00
ran_0050_00075	0.00 (5)	0.00 (5)	0.00 (5)	48.09	1.31	0.41	0.33	0.00
ran_0050_00100	0.00 (5)	0.00 (5)	0.00 (5)	76.52	2.13	0.34	0.21	0.00
ran_0050_00200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	0.60	0.14	0.13	0.00
ran_0050_00500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	2.05	0.15	0.14	0.00
ran_0050_00750	0.00 (5)	0.00 (5)	0.00 (5)	100.00	2.38	0.16	0.16	0.00
ran_0100_00150	0.00 (5)	0.00 (5)	0.00 (5)	47.86	110.73	3.82	2.49	0.00
ran_0100_00200	2.00 (4)	0.00 (5)	0.00 (5)	68.94	115.86	5.10	4.63	0.00
ran_0100_00400	0.00 (5)	0.00 (5)	0.00 (5)	100.00	3.67	0.20	0.17	0.00
ran_0100_01000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	7.76	0.22	0.22	0.01
ran_0100_01500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	13.12	1.15	1.15	0.03
ran_0300_00450	14.69 (0)	0.00 (5)	0.00 (5)	45.79	–	653.22	294.31	0.01
ran_0300_00600	16.50 (0)	0.59 (4)	0.00 (5)	63.83	–	456.20	789.24	0.02
ran_0300_01200	0.00 (5)	0.00 (5)	0.00 (5)	100.00	37.83	4.96	1.36	0.06
ran_0300_03000	0.00 (5)	0.00 (5)	0.00 (5)	100.00	50.23	3.92	1.50	0.15
ran_0300_04500	0.00 (5)	0.00 (5)	0.00 (5)	100.00	471.89	0.99	0.47	0.23
ran_0500_00750	98.48 (0)	1.81 (1)	1.78 (1)	45.79	–	1484.69	1500.73	0.03
ran_0500_01000	88.56 (0)	8.27 (0)	2.64 (1)	62.33	–	–	1311.19	0.04
ran_0500_02000	6.67 (4)	0.00 (5)	0.00 (5)	99.13	406.83	19.53	1.52	0.14
ran_0500_05000	40.00 (3)	0.00 (5)	0.00 (5)	100.00	618.27	6.63	2.83	0.40
ran_0500_07500	20.00 (4)	0.00 (5)	0.00 (5)	100.00	545.82	2.55	2.54	0.62
ran_1000_01500	96.90 (0)	5.80 (0)	5.20 (0)	43.07	–	–	–	0.15
ran_1000_02000	98.81 (0)	14.38 (0)	6.27 (0)	62.00	–	–	–	0.22
ran_1000_04000	100.00 (0)	0.00 (5)	0.00 (5)	99.68	–	108.74	6.43	0.49
ran_1000_10000	60.00 (2)	0.00 (5)	0.00 (5)	100.00	910.21	12.49	6.40	1.80
ran_1000_15000	40.00 (3)	0.00 (5)	0.00 (5)	100.00	1266.90	4.76	3.83	3.19
Sol. opt.	**41.33%**	**73.33%**	**76.00%**		**454.39**	**134.52**	**146.74**	**0.51**

Chapitre 5

Conclusion générale

5.1 Conclusions et perspectives scientifiques

5.1.1 L'essor des réseaux

La problématique de l'optimisation dans les réseaux n'est pas nouvelle. Elle a toujours reçu beaucoup d'attention comme en témoigne l'existence du *Network Simplex* et l'abondante bibliographie disponible. Les progrès de l'électronique ont rendu possible la constitution de réseaux de communication plus en plus nombreux, et de taille croissante, l'Internet constituant sans doute l'exemple le plus emblématique. Tout porte à croire que cette tendance se poursuivra comme l'annoncent les tenants de l'informatique ambiante (*ambient computing*), ce qui rendra l'optimisation des réseaux de plus en plus souhaitable, qu'il s'agisse de la qualité de service (bande passante, zones couvertes), ou de durée de vie lorsque les nœuds du réseaux sont constitués d'appareils mobiles. Dans ce contexte, la taille des problèmes d'optimisation augmentant, il me semble que la combinaison des méthodes heuristiques et des approches exactes peut permettre, dans une certaine mesure, de répondre aux besoins futurs.

5.1.2 Aide à la décision

Les gestionnaires ou les concepteurs de ces réseaux se verront sans doute confrontés à des problèmes de décision dont les enjeux et la complexité nécessiteront une aide à la décision à la hauteur de ces nouveaux réseaux. A cet égard, les travaux présentés au chapitre 3 constituent une sorte de « degré zéro » de l'aide à la décision, car l'aide en question se résume souvent à éclairer le décideur sur les conséquences de ses choix. Cette acception minimaliste de l'aide à la décision est aussi le résultat de mon expérience avec les industriels (de ST Microelectronics en 2001 à Bolloré Plastic Film Division en 2012). La prise de décision reste souvent une prérogative humaine, notamment lorsque les enjeux sont importants. Cela s'explique aussi par le fait que la décision prise *in fine* fait intervenir un grand nombre de paramètres (techniques, commerciaux, stratégiques) qu'il est difficile d'appréhender, ne serait-ce que parce qu'ils sont changeants, et souvent non explicités. Les industriels m'ont souvent paru se satisfaire d'un outil permettant d'éclairer leurs décisions, et ne m'ont jamais semblé être demandeurs de systèmes permettant de prendre des décisions « à leur place ». Cette défiance vis-à-vis des outils logiciels se retrouve aussi dans l'usage qui est fait des outils mis à leur disposition (les utilisateurs essayant de faire mieux que le logiciel), et le souhait de comprendre le cheminement ayant conduit à proposer une solution reste très fort. La mise en œuvre immédiate de méthodes plus avancées relevant de l'aide à la décision [61] me paraît difficile, et l'expérience me fait préférer une approche plus progressive (d'abord un logiciel « qui éclaire » les décisions humaines, ensuite une approche qui suggère des décisions). Le maître mot de la réussite est assurément la confiance entre ceux qui développent les outils, et ceux qui les exploitent.

5.1.3 Une approche pluridisciplinaire

La confiance des industriels se gagne peut-être plus facilement si les chercheurs développant des méthodes d'aide à la décision sont capables d'appréhender leurs problématiques, les forces et les faiblesses de leur outil de production (en termes de flexibilité, de flux poussé ou de flux tiré). Ma formation d'ingénieur en automatique et productique m'a récemment permis de faire apparaître que les gisements d'efficacité dans ce qui s'apparentait au départ à un problème de découpe, n'étaient pas nécessairement là où on les cherchait initialement, en raison de l'impact de la gestion des approvisionnements et des ordres de fabrication dans l'entreprise concernée. Convaincu qu'un scientifique est d'abord quelqu'un qui sait replacer les problèmes dans leur contexte et dont l'esprit critique l'incite à questionner la pertinence d'hypothèses implicites présentés comme évidentes de prime abord, je vois la pluridisciplinarité comme une caractéristique du chercheur opérationnel, dont la discipline scientifique n'est pas liée à un domaine d'application particulier, mais se trouve au contraire au carrefour de nombreuses autres disciplines. Outre la question de sa formation initiale, cette démarche me semble devoir conduire le chercheur opérationnel à rester ouvert aux autres disciplines, s'il veut découvrir de nouveaux problèmes pertinents dont il peut comprendre la portée, évitant ainsi de voir son rôle confiné à celui d'un expert en recettes mathématiques ou informatiques dépourvu de vision globale.

5.1.4 Développement des matheuristiques

Ma collaboration avec Alok Singh, dont l'approche des problèmes est très différente de la mienne, m'a convaincu que le succès des méthodes exactes, comme des métaheuristiques repose avant tout sur une bonne compréhension des problèmes. En travaillant plus étroitement avec les membres de la communauté des métaheuristiques, j'ai pris conscience que les algorithmes les plus efficaces étaient très souvent ceux pour lesquels la structure du problème

était exploitée judicieusement. J'y vois là encore une caractéristique commune aux travaux de tous les chercheurs opérationnels. L'énoncé des problèmes classiques est souvent d'une simplicité déconcertante (problème du sac à dos, de la coloration des sommets d'un graphe), de sorte qu'on peut être porté à penser qu'il n'y a rien de particulier à comprendre sur le problème lui-même. La littérature montre qu'il n'en est rien (on peut penser aux inégalités *comb* pour le problème du voyageur de commerce). Or si les publications orientées métaheuristiques énoncent peu de résultats théoriques, on ne peut faire l'économie de cette connaissance approfondie (et donc non évidente) du problème étudié. L'établissement de propriétés (cas particuliers où le problème est facile, symétries, etc) que l'on trouve plus fréquemment dans les travaux des chercheurs opérationnels développant des approches de résolution exactes, peut se révéler très utile pour proposer une métaheuristique. Ce rapprochement se fait naturellement lorsqu'on développe des matheuristiques, qui exploitent conjointement des méthodes exactes et approchées.

5.2 Réflexions sur la direction de recherches

Diriger des recherches consiste à donner une direction, ce qui suppose une réflexion sur la recherche scientifique, et le contexte dans lequel la direction en question peut s'exercer. Je distingue ici trois types de directions de recherche. D'abord, celle qui relève des choix individuels, et qui concerne tout doctorant, chercheur ou enseignant-chercheur. Ensuite, bien sûr, la direction de thèse, qui revêt plus de responsabilité parce qu'elle implique autrui, et renvoie celui qui l'exerce à l'orientation de sa propre recherche, ainsi qu'aux valeurs qu'il transmet aux jeunes chercheurs qu'il a la responsabilité de former. Enfin, la direction d'une équipe de recherche, qui se distingue des précédentes formes de direction par une plus grande interaction avec le laboratoire, son organisation, sa stratégie et sa gouvernance, et qui revêt un rôle social impliquant des responsabilités plus étendues.

5.2.1 La direction de sa propre recherche

Le décret numéro 84-431 du 6 juin 1984 renvoie à l'article L112-1 du Code de la recherche, qui stipule que le premier objectif de la recherche publique est « le développement et le progrès de la recherche dans tous les domaines de la connaissance ». Je veux montrer ici ma vision de la portée que revêt la direction de la recherche, les limites que je perçois et sous quelles contraintes les choix d'un universitaire en matière de recherche peuvent s'opérer.

Evaluation de la recherche et production scientifique

L'évaluation d'un enseignant-chercheur se focalise presqu'exclusivement sur sa production scientifique et dans une moindre mesure sur sa contribution à l'animation de réseaux (organisation de conférences, d'événements, direction de projets, responsabilités administratives). Cet état de fait conduit souvent les enseignants-chercheurs à se focaliser sur des questions comme « que publier et dans quel journal ou quelle conférence ». Cette focalisation peut s'opérer au détriment de questions plus fondamentales comme « sur quels thèmes faut-il exercer ses activités de recherche, et comment organiser son travail », comme si ces questions étaient moins importantes ou avaient une réponse évidente. A mon sens, il n'en est rien.

L'AERES considère qu'un enseignant-chercheur en sciences et technologies de l'information et de la communication est « produisant en recherche et valorisation » (le terme « publiant » a semble-t-il été banni de la terminologie officielle), s'il publie au moins deux articles dans des revues de rang A par quadriennal [21]. Notons au passage qu'il n'existe aucune définition faisant autorité de ce qu'est exactement une « revue de rang A ». Bien que l'AERES propose une définition dans [21], il apparaît que chaque communauté scientifique ou presque a sa propre définition. On retrouve ce besoin de classer les revues à un niveau de granularité encore plus fin : le Lab-STICC a par exemple établi son propre classement des revues, à l'usage de ses membres.

Cette profusion de classements trouve probablement son origine dans l'impossibilité d'établir une mesure objective et universelle de la qualité d'un travail de recherche, aussi chacun prend-il soin d'établir un classement conforme à sa perception. La confusion qui entoure la question du classement des journaux scientifiques est analogue à celle qui prévaut en matière de classement mondial des universités. Jean-Charles Billaut, Denis Bouyssou et Philippe Vincke ont mis en évidence les insuffisances du célèbre classement de Shangaï, ainsi que la difficulté inhérente à tout classement de ce type [11]. Par ailleurs, Denis Bouyssou et Thierry Marchant ont montré que le classement des chercheurs et des institutions basé sur le *h-index* avait un fondement scientifique contestable.

Stratégie de recherche et de publication

Dans ce contexte, un maître de conférences en début de carrière aura à cœur de se conformer le plus rapidement possible à cet impératif de publication, fût-il mal formulé, pour éviter d'être étiqueté « non publiant ». Cette

pression, très forte, conduit à privilégier les activités de recherches susceptibles de permettre de publier rapidement dans des revues dont on espère qu'elles seront considérées comme étant de rang A par la direction du laboratoire, le conseil scientifique de l'université, l'AERES, le CNRS, les experts chargés de classer les demandes de PES, les membres du CNU qui statuent sur les demandes de qualification, et *in fine* les comités de sélection. Cela fait du monde, et tous ces évaluateurs ne partagent pas nécessairement la même appréciation d'une revue scientifique donnée. Il en ressort que la meilleure stratégie semble être celle qui conduit à l'obtention de résultats publiables dans les délais les plus courts, sans faire de compromis sur la qualité des revues. Comme on a davantage de chances de succès quand on publie dans un domaine que l'on connaît déjà, cette course aux publications ne favorise guère la mobilité thématique. Elle conduit même certains à exploiter un filon, voire à se spécialiser dans une niche. Ces deux métaphores, très souvent employées, sous-entendent une certaine forme de repli qui me paraît préjudiciable. A court terme, il suffit que le produit du filon passe de mode pour que celui qui l'exploite ne puisse plus rien en tirer. Mais plus fondamentalement, science et logique d'exploitation me paraissent incompatibles. Peut-on progresser sans remise en question, sans prendre le risque de se confronter à de nouveaux problèmes, sans s'approprier de nouveaux savoirs ?

Charles de Gaulle, alors Président de la République, aurait déclaré : « des chercheurs qui cherchent, on en trouve. Des chercheurs qui trouvent, on en cherche » [35]. Cela me semble traduire le fait que pour le dirigeant, le financeur, le résultat de la recherche prime sur la recherche elle-même. La citation suivante, attribuée à Albert Einstein semble répondre à celle du général : « Pour trouver sans chercher, il faut avoir longtemps cherché sans trouver ». Cela implique que pour devenir un bon chercheur, il est probablement nécessaire d'avoir travaillé un certain temps sans obtenir de résultats valorisables à court terme. Cette période, bien qu'improductive si l'on réduit la productivité d'un chercheur à son seul aspect bibliométrique, est pourtant d'une importance capitale : tout chercheur a vu un jour une de ses idées n'ayant pas permis de résoudre un problème particulier se révéler d'une efficacité décisive pour résoudre un autre problème quelques mois ou quelques années plus tard. Le fait de travailler sur plusieurs problèmes différents me paraît favoriser l'éclosion d'idées nouvelles qui, même si elles ne démontrent pas tout leur potentiel immédiatement, sont fécondes (Henri Poincaré disait : « Ce qui vous frappera d'abord, ce sont les apparences d'illumination subite, signes manifestes d'un long travail inconscient ; le rôle de ce travail inconscient dans l'invention mathématique me paraît incontestable »). Mais pour que ces idées nouvelles émergent, il faut pouvoir consacrer du temps à ses recherches, et se soustraire au moins temporairement à l'impératif des résultats publiables à court terme. A cet égard, mes séjours à l'université d'Hyderabad ont été autant de parenthèses salutaires.

Concilier objectifs personnels et institutionnels

Il reste une question dont je n'ai pris la mesure de l'importance que trop tardivement : le choix des thèmes de recherche. A l'inverse du système académique indien, qui laisse une liberté thématique quasi-totale à ses enseignants-chercheurs, la structuration de la recherche en France est plus rigoureuse, car les laboratoires sont incités à positionner leurs activités autour d'un nombre restreint de points forts. L'objectif visé est d'accroitre leur visibilité au plan international et d'atteindre l'excellence sur les thématiques retenues (il n'est pas interdit de penser que le classement de Shangaï y a également sa part). Le recrutement d'un enseignant-chercheur s'inscrit donc dans cette politique, et on attend d'un nouveau membre du laboratoire qu'il renforce une ou plusieurs des thématiques prioritaires. Cela sous-entend que dans bien des cas, le choix des thèmes de recherche d'un enseignant-chercheur a déjà été arrêté pour lui, comme en témoignent les profils de poste d'enseignants-chercheurs parfois extrêmement précis. Intégré à une équipe de recherche, le nouveau maître de conférences n'a plus qu'à se conformer au projet d'intégration qu'il a présenté au comité de sélection, et sur la base duquel sa candidature a été retenue. Comme le choix des thématiques prioritaires du laboratoire nécessite une vision globale du monde académique et des orientations stratégiques des divers organismes de recherche (localement l'université, le département et la région, et au plan national le CNRS et le ministère de l'enseignement supérieur), un maître de conférences est généralement peu concerné par ces questions. C'est regrettable car elles sont fondamentales, et ont même indirectement un impact très important sur lui.

En effet, les orientations prises par un laboratoire peuvent avoir des motivations scientifiques, sociétales ou politiques qu'il est parfois difficile de faire coïncider avec les critères académiques utilisés pour évaluer ses membres. Ainsi, le conseil régional peut légitimement désirer voir privilégier certaines thématiques de recherche par un laboratoire implanté sur son territoire, alors même qu'il est très difficile de publier dans des revues de rang A du domaine en question. Certains thèmes peuvent être également perçus comme « passés de mode » par la communauté académique, rendant la publication des travaux particulièrement difficile. Ces considérations me conduisent à penser qu'il existe un écart très important entre la mission de recherche des enseignants-chercheurs telle qu'elle est définie par les textes réglementaires, et l'exercice effectif de cette recherche, qui les conduit à composer avec la politique des organismes dont ils dépendent (laboratoire, école d'ingénieurs, UFR, CNRS, Ministère de l'enseignement supérieur), et les instances d'évaluation tout aussi nombreuses.

La place de la recherche dans le métier d'enseignant-chercheur

La multiplication quelque peu erratique de ces évaluations, le caractère versatile et parfois contradictoire des critères retenus ne sont finalement que le reflet des changements profonds qui traversent la société toute entière. Il n'en reste pas moins vrai que ces évaluations n'ont finalement pour véritable objectif que de vérifier que les enseignants-chercheurs s'acquittent correctement de leurs missions, missions que l'article 3 du décret 84-431 du 6 juin modifié le 1er septembre 2009 a considérablement élargies. A mon sens, cet élargissement ne doit pas être interprété comme une injonction de couvrir tout le spectre des missions mentionnées par le décret, mais plutôt comme une plus grande liberté offerte aux enseignants-chercheurs, leur permettant de se consacrer à des activités différentes au cours de leur carrière, à l'heure où celles-ci s'allongent au rythme de l'espérance de vie [28]. A titre personnel, je reste très attaché aux missions historiques de ce métier : l'enseignement (ce point ne sera pas développé ici) et la recherche. Les invitations à séjourner dans les universités étrangères, les CRCT, détachements et mises en disponibilité me semblent essentielles pour se ménager des moments de « respiration », où l'on peut reprendre sa liberté thématique, en même temps que du recul sur son établissement de rattachement, voire sur son pays lorsqu'on a l'opportunité de voyager au-delà des frontières françaises. Comme il est évident qu'aucun système d'organisation de la recherche n'est parfait (même si l'herbe est toujours plus verte ailleurs), j'approuve l'idée défendue par Hamming dans son discours *You and your research* [36] selon laquelle il est vain de s'opposer frontalement au système au motif qu'il est imparfait, et qu'il vaut mieux au contraire s'en accommoder du mieux qu'on peut. Il offre heureusement suffisamment de liberté pour trouver les marges de manœuvre permettant de satisfaire les enseignants-chercheurs, les institutions auxquelles ils appartiennent, et les organes d'évaluation.

5.2.2 La direction de thèse

La formation à la recherche et par la recherche

En matière de formation doctorale, la notion de formation à la recherche et par la recherche, que mentionne l'article 3 du décret 84-431 du 6 juin 1984 modifié le 1er septembre 2009 m'a longtemps laissé perplexe. J'y ai d'abord vu à la fois un truisme et une contradiction. Un truisme parce que toute activité, tout métier, s'apprend et surtout se perfectionne par la pratique. Une contradiction parce que la simplicité de son énoncé tranche singulièrement avec la pratique quotidienne de la recherche scientifique.

Je souhaite ouvrir cette section sur la direction de thèse par l'évocation de mes premiers pas en recherche, lors de mon stage de DEA. Du jour au lendemain, il fallait passer de la salle de classe à la solitude d'une salle des doctorants, qui se trouvait être vaste et déserte. Le fond et la forme du travail changent également du tout au tout, les cours disparaissant au profit de la lecture et de l'analyse d'articles. Ce choc fut à la mesure de celui qu'il me faudrait affronter moins d'un an plus tard : encadrer une séance de TP pour la première fois. Dans un cas comme dans l'autre, l'impression de sauter dans l'inconnu est particulièrement forte, et le sentiment d'avoir été mal préparé ne l'est pas moins. Heureusement, la transition est rapide.

Le rôle de l'encadrement de thèse est primordial à cet égard. Pour ce qui est de l'enseignement, un parallèle peut être fait dans une certaine mesure avec le tutorat mis en place par le CIES pour ses moniteurs, mais cet aspect du métier d'enseignant-chercheur ne sera pas traité ici. Les encadrants de thèse sont souvent les seuls interlocuteurs du doctorant, et à ce titre ils occupent le rôle principal du schéma de la formation des doctorants à la recherche, par la recherche. Ceci est encore plus vrai pour les doctorants en contrat CIFRE, qui passent moitié moins de temps au laboratoire que les autres.

Après seulement trois années de pratique de la recherche, je devenais à mon tour co-encadrant de master recherche. Là encore avec le sentiment que les choses étaient allées vite, sans pour autant ressentir l'impression d'impréparation avec la même intensité. Un an après l'obtention de mon doctorat, je me voyais proposer mon premier co-encadrement de thèse, que j'abordais cette fois-ci sans appréhension, la formation à la recherche et par la recherche ayant manifestement porté ses fruits malgré ma perplexité initiale. Plus spécifiquement, c'est ma directrice de thèse qu'il convient de remercier. Cette première expérience de la recherche et la pratique de l'encadrement doctoral de Mireille Jacomino n'ont cessées d'être une source d'inspiration pour moi. Si les années qui ont suivi m'ont beaucoup apporté, et donné l'opportunité d'affiner et de personnaliser la manière dont j'appréhende l'encadrement doctoral, les fondamentaux restent ceux que Mireille Jacomino m'a enseignés par son exemple. Aujourd'hui, même si la formulation ne me satisfait pas, il m'apparaît que la formation à la recherche et par la recherche est sans doute le meilleur système de formation à la recherche. Il ne doit cependant son efficacité qu'au dévouement et au professionnalisme de celles et de ceux qui le mettent en œuvre. Les sections qui suivent présentent ce qui, à mon sens, conditionne la réussite de ce modèle de formation des doctorants.

Les valeurs de la direction de thèse

Les règles de bonne conduite relatives aux publications (probité intellectuelle, respect de la propriété intellectuelle et refus du plagiat) font partie intégrante des valeurs du chercheur. Ces règles me paraissent bien connues, ne serait-ce que parce qu'elles sont écrites (sur le site web des éditeurs de journaux scientifiques) et que chacun est explicitement appelé à s'y conformer lors de la soumission d'un article. Je n'en dirai donc rien de plus ici, sinon que les encadrants de thèse doivent veiller à ce que leurs doctorants aient connaissance de ces règles.

C'est sur les murs de l'école primaire du campus d'Hyderabad que j'ai trouvé, parmi d'autres maximes inscrites à la peinture, la formulation d'une des premières qualités d'un encadrant de thèse : *Who dares teaching must never cease to learn.* Je suis fermement convaincu qu'un bon encadrant de thèse doit d'abord être un modèle de ce qu'est un bon chercheur, et ne pas se contenter de transmettre un savoir ou de dispenser des conseils, si précieux soient-ils, car le doctorant apprend beaucoup par l'exemple. Ma directrice de thèse entamait elle-même une reconversion thématique alors qu'elle encadrait mes travaux de thèse. Sa grande ouverture d'esprit, son esprit critique, sa volonté de progresser dans un domaine nouveau, sa façon d'aborder un nouveau problème m'ont inspiré pour les avoir vus en œuvre à de nombreuses occasions. C'est cet enseignement sur l'état d'esprit du chercheur qui est à l'origine de mon aversion pour le cantonnement de l'activité de recherche à une niche et à l'exploitation de filons. Il s'agit plus généralement de la méfiance à l'égard de la facilité, qui n'est sans doute pas la meilleure conseillère du chercheur. Concernant plus spécifiquement le co-encadrement de thèse, la loyauté est également une valeur que tout chercheur doit partager : les encadrants ne doivent pas afficher leurs désaccords en présence de leur doctorant, et une fois que chacun s'est exprimé, le dernier mot revient au directeur de thèse. Le co-encadrant doit s'efforcer de se conformer à la stratégie choisie par le directeur de thèse, même s'il ne l'approuve pas totalement (ce dont il doit pouvoir discuter librement avec le directeur de thèse, mais en privé). Je n'ai pris conscience des divers aspects éthiques du métier d'enseignant-chercheur qu'assez tardivement, considérant alors que les questions d'éthique étaient l'apanage des chercheurs en sciences du vivant, du nucléaire ou des sciences sociales. C'est à Colin de la Higuera, dont j'ai découvert les présentations par hasard (à la faveur d'un e-mail adressé à tout le laboratoire), que je dois cette découverte enrichissante.

La direction de thèse au quotidien

L'artiste au sommet de son art laisse souvent aux spectateurs une impression d'aisance, qui masque la difficulté liée à la pratique. Ce constat s'est souvent imposé à moi lors de mes débuts comme co-encadrant de thèse. Je veux parler ici plus spécifiquement de l'équilibre à trouver entre la liberté d'initiative du doctorant, et la nécessité de donner une direction à ses travaux. Ce savant dosage, que ma directrice de thèse avait trouvé pour moi, n'était bien sûr pas transposable tel quel. Et malheureusement, le souvenir de ma propre thèse ne m'a pas incité à reconsidérer cet équilibre immédiatement. Fort heureusement, le co-encadrement offre la possibilité de voir à l'œuvre d'autres approches de l'encadrement doctoral, et de ne pas persister dans l'erreur. Une autre origine de cette erreur d'appréciation est un *a priori* consistant à supposer que tout étudiant qui entreprend une thèse le fait pour les mêmes motifs que soi. La compréhension de ces motivations est d'autant moins aisée que celles-ci peuvent évoluer au cours du temps, en fonction notamment de l'idée que le doctorant se fait de son niveau, de sa place dans la communauté scientifique, et de sa perception de ses chances d'accéder à un poste d'enseignant-chercheur en France ou dans son pays d'origine s'il est étranger. J'ai été surpris que ces questions, pourtant fondamentales, ne soient presque jamais abordées à l'initiative du doctorant. Lorsqu'elles le sont à l'initiative de ses encadrants, il faut veiller à ce que l'apparition de ce sujet ne soit pas perçue par le doctorant comme l'expression d'une inquiétude de ses encadrants quant à ses capacités. En effet, eu égard au contexte de l'emploi public en France, il faut dire clairement et honnêtement la difficulté de trouver un poste de maître de conférences ou de chargé de recherches, et ce discours est souvent perçu comme décourageant par les intéressés, ce qui est bien évidemment contre-productif. On voit alors un certain nombre d'entre eux se tourner vers l'industrie sans avoir épuisé toutes les options qui s'offrent aux nouveaux docteurs (postes d'ATER, stages post-doctoraux). Bien sûr, cela peut s'expliquer par la précarité de ces postes, les salaires plus élevés pratiqués dans le secteur privé, et les motivations personnelles de chacun. Il me semble cependant que l'attractivité des activités d'ingénierie tient au moins en partie au nouveau mode d'organisation de la recherche dans les laboratoires, dont je reparlerai plus loin : la généralisation des projets conduits par les universitaires et les industriels. Ces projets sont devenus une source de financement d'importance croissante pour les thèses. Or dans certains cas, la dimension prise par la recherche et développement et le transfert de technologie dans ces projets a conduit à financer des thèses où les activités d'ingénierie ont quelque peu éclipsé la recherche scientifique. A l'instar des anciens doctorants sur contrat CIFRE, les docteurs ayant pris part à ces projets se tournent peut-être plus naturellement vers l'ingénierie. Dans ce contexte, le directeur de thèse et le chef de projet se confondent aux yeux du doctorant, et celui-ci doit veiller à ne pas être qu'un chef de projet, afin que la formation à la recherche et par

la recherche ne soit pas vidée de son sens.

5.2.3 La direction d'équipe de recherche

Colin de la Higuera, dans un diaporama sur le networking [37], fait remarquer qu'assister à des présentations en conférence permet de tirer de précieux enseignements sur ce qu'il faut faire, mais aussi sur ce qu'il ne faut pas faire. A défaut d'avoir été moi-même chef d'équipe, je me base sur mon expérience de membre d'équipe pour présenter ce qui, de mon point de vue, fait qu'une équipe de recherche fonctionne bien ou pas. Après avoir présenté le contexte particulier de mes observations, j'aborderai plus spécifiquement le rôle du responsable ou du chef d'équipe.

J'ai été membre de l'équipe Commande Robuste Surveillance et Supervision (6 permanents) pendant mon séjour au LAG en tant que stagiaire de master, puis doctorant, et enfin ATER. Au LESTER, j'ai été membre de l'équipe Systèmes reconfigurables, de taille plus modeste (3 permanents), et aujourd'hui je suis membre de l'équipe Recherche opérationnelle, qui compte seulement deux permanents (Marc Sevaux et moi). Ce cas particulier ne relève pas d'une équipe à proprement parler, car les autres membres sont des doctorants que nous co-encadrons, aussi les observations que je ferai sur le thème de la direction d'équipe sont-elles le fruit d'observations plus anciennes, voire du récit de situations dont je n'ai pas été le témoin direct, et ne s'appliquent pas à l'équipe Recherche opérationnelle.

La délicate mécanique de l'adhésion

Tout d'abord, j'ai observé que la notion même d'équipe était de plus en plus vivement contestée. J'ai vu plus souvent les équipes de recherche se dissoudre que se constituer. Il me faut cependant préciser que les laboratoires où j'ai travaillé ont toujours été à un moment ou à un autre, le théâtre de restructurations profondes, qui j'en suis persuadé, ne peuvent qu'affaiblir la cohésion des équipes de recherche. Ces restructurations étaient toujours conduites avec comme objectif affiché de favoriser la constitution de nouvelles équipes autours de projets. Autrement dit, des équipes temporaires viennent se juxtaposer aux équipes de recherche préexistantes, qui avaient été fondées en suivant une logique thématique. Il s'agit je crois de la transposition aux laboratoires de l'organisation matricielle aujourd'hui très répandue dans l'entreprise.

Bien souvent, j'ai observé que les équipes de recherche « historiques » n'avaient que rarement survécu à ces changements radicaux d'organisation. Par ailleurs, ces restructurations se sont toujours accompagnées de l'émergence d'exigences sur la performance individuelle des chercheurs (cf. les chercheurs publiants et les autres, qui se voient menacés à terme d'être chassés du laboratoire). Cela conduit à mon sens à une individualisation de l'exercice de la recherche. Chacun monte ses projets, de préférence avec des partenaires académiques ou industriels hors du laboratoire. Pour peu que les équipes de recherche « historiques » ne soient pas le siège d'une collaboration étroite et effective, leurs membres désertent progressivement les réunions et se détachent de l'équipe.

Dans ce contexte, le rôle du responsable d'équipe revêt une importance plus grande encore que par le passé, car l'équipe qu'il dirige est désormais en concurrence avec la nouvelle organisation « par projet » qu'encourage fortement la direction du laboratoire. La cohésion de son équipe n'est plus naturelle ni acquise. Il lui faut la rendre attractive s'il ne peut retenir ses membres par la contrainte. De fait, il m'a semblé que les équipes dirigées par des chercheurs se distinguant par leur envergure ou leur force de caractère étaient moins souvent sujettes à l'éclatement. Si un chef d'équipe ne peut pas retenir les membres de son équipe en usant de sa seule autorité, il peut y créer des conditions de travail telles que ses membres ont objectivement plus d'intérêt à y adhérer qu'à s'en détacher.

L'autorité du chef d'équipe découle de son rayonnement ou de la reconnaissance de son travail dans la communauté scientifique à laquelle il appartient. Si les membres de son équipe n'appartiennent pas à la même communauté scientifique que lui, ceux-ci penseront peut-être qu'ils ont peu à apprendre de leur chef, et seront tentés de quitter l'équipe s'ils ne trouvent pas d'intérêt particulier à prendre part aux réunions.

La direction d'équipe au quotidien

Fort de ce constat, je pense qu'un bon chef d'équipe doit toujours garder à l'esprit cette mécanique de l'adhésion. Aussi ne doit-il pas considérer que son rôle se limite à organiser des réunions, il doit être un véritable animateur scientifique, et veiller à ce que chacun tire profit de son appartenance à l'équipe de recherche qu'il dirige. Le moyen le plus naturel d'y parvenir me semble être de monter des projets impliquant les membres de l'équipe. Ainsi, sa légitimité se trouve renforcée par son rôle de chef de projet, qui lui permet de se situer simultanément sur les deux dimensions de la matrice hiérarchique du laboratoire.

Pour ce qui concerne son action au quotidien, il me semble qu'il est également de son devoir de se soucier du devenir des membres temporaires de son équipe : les doctorants. En premier lieu, il doit être le relai entre la politique du laboratoire, et ceux qui la mettent en œuvre quotidiennement. Au cours des réunions qu'il anime, il est bon qu'il

explique où va le laboratoire, où va l'équipe, quels sont les projets qu'elle porte, les objectifs qu'elle vise et quel est le rôle de chacun. Cela permet aux doctorants de bien se situer dans le laboratoire, d'en comprendre les objectifs stratégiques et la gouvernance, et leur ouvre des perspectives dépassant le périmètre de leurs encadrants. L'équipe de recherche devrait également permettre aux doctorants d'éprouver leurs prestations orales avant de présenter leurs travaux en conférence. Une opinion sincère de tous les membres de l'équipe est d'autant plus précieuse qu'il est assez rare de recevoir des conseils sur la forme de ses prestations en conférence. Enfin, une fois la thèse soutenue, le chef d'équipe et tous ses membres devraient aider activement le nouveau docteur à trouver un stage post-doctoral, car tout ce qui fait la force d'une équipe n'est pas seulement à l'intérieur, elle existe aussi à travers son réseau, et le regard que lui porte le monde extérieur (le post-doctorant se souviendra toujours de ce qu'il doit à son ancienne équipe de recherche). Cette aide est fournie d'autant plus facilement que chacun dans l'équipe connaît assez bien les travaux de l'ancien doctorant pour avoir participé, même modestement, à sa formation en le conseillant de temps à autre. Bien sûr, le même type de solidarité devrait s'étendre à la rédaction d'articles, où un doctorant me semble raisonnablement pouvoir demander à tel ou tel membre de l'équipe un avis sur un résumé ou une section particulière d'un de ses articles. Au delà des doctorants, le chef d'équipe devrait aussi se préoccuper de la carrière des maîtres de conférences et leur permettre de bénéficier de son expérience dans la rédaction d'articles, la stratégie de publication, la préparation de dossiers de qualification, le montage de projet.

Conclusion

Au début de ma carrière, l'enseignement, la recherche et les charges administratives me semblaient être des activités totalement différentes. Avec le temps et la prise de responsabilités, elles me sont apparues de plus en plus semblables, non pas dans la forme mais dans la manière de les appréhender. Elles requièrent toutes des qualités d'organisation, de cohérence, d'exigence envers soi-même, et la nécessité de susciter l'adhésion. Pour ce qui concerne plus particulièrement la recherche, il est naturel que la prise croissante de responsabilités se traduise par une certaine diminution de la participation à des travaux scientifiques en tant que contributeur principal. Il me paraît cependant primordial de rester au plus près des activités de base du chercheur, même de celles qui passent pour les plus ingrates, comme la programmation. Cela me semble offrir la garantie de ne pas perdre de vue la difficulté de ces tâches (qui sont souvent déléguées aux doctorants), et de ne pas se couper du cœur du métier d'enseignant-chercheur. Si le surcroît de responsabilités qu'une habilitation à diriger des recherches permet d'endosser a nécessairement pour corolaire un positionnement différent de ma contribution scientifique, je souhaite rester un « chercheur qui cherche », dans un cadre désormais plus collectif, et un jour prochain assumer la responsabilité d'une « équipe qui trouve ».

Bibliographie

[1] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows : Theory, Algorithms, and Applications*. Van Nostrand Reinhold, New Jersey : Prentice Hall, 1993.

[2] I.F. Akylidiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor network. *IEEE Communication Magazine*, 40(8) :102–116, 2002.

[3] A. Alfieri, A. Bianco, P. Brandimarte, and C.F. Chiasserini. Maximizing system lifetime in wireless sensor networks. *European Journal of Operational Research*, 181 :390–402, 2007.

[4] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron, and L. Liberti. Column generation algorithms for exact modularity maximization in networks. *Physical Review. E*, 82 :046112.1–046112.9, 2010.

[5] A. Arora and S. Subramaniam. Wavelength conversion placement in wdm mesh optical networks. *Photonic Network Communications*, 4 :167–177, 2002.

[6] A. Aubry, M-L. Espinouse, and M. Jacomino. A max-min approach to delay load-shedding in power distribution networks despite source-capacity uncertainties. In *Proceedings of the International Network Optimization Conference*, pages 1–6, 2007.

[7] P. Avella, D. Villacci, and A. Sforza. A steiner arborescence model for the feeder reconfiguration in electric distribution networks. *European Journal of Operational Research*, 164 :505–509, 2005.

[8] P. Baptiste, E. Néron, and F. Sourd. *Modèles et algorithmes en ordonnancement*. Ellipses, 2004.

[9] L. Benini, G. Castelli, A. Macii, E. Macii, M. Poncino, and R. Scarsi. A discrete-time battery model for high-level power estimation. In *Proceedings of the IEEE Design Automation and Test in Europe conference*, pages 35–39, Paris, France, 2000.

[10] L. Benini, D.Bruni, A. Macii, E. Macii, and M. Poncino. Discharge current steering for battery lifetime optimization. *IEEE Transaction on computers*, 52(8) :985–995, 2003.

[11] J-C. Billaut, D. Bouyssou, and P. Vincke. Faut-il croire le classement de Shangaï ? *Revue de la régulation*, 8 :1–44, 2010.

[12] R. Borndörfer, U. Schelten, T. Schlechte, and S. Weider. *Column generation approach to airline crew scheduling*, pages 343–348. Springer, 2005.

[13] M. Cardei, M.T. Thai, Y. Li, and W. Wu. Energy-efficient target coverage in wireless sensor networks. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1976–1984, 2005.

[14] R. Cerulli, M. Gentili, and A. Iossa. Bounded-degree spanning tree problems : models and new algorithms. *Computational Optimization and Applications*, 42 :353–370, 2009.

[15] M. X. Cheng, L. Ruan, and W. Wu. Achieving minimum coverage breach under bandwidth constraints in wireless sensor networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 2638–2645, 2005.

[16] V. Chvátal. *Linear Programming*. W. H. Freeman and Company, New York, 1980.

[17] M. Crappe. *Stabilité et sauvegarde des réseaux électriques*. Hermès Sciences, Paris, 2003.

[18] D. Das. A fuzzy multiobjective approach for network reconfiguration of distribution systems. *IEEE Transaction on Power Delivery*, 21 :202–209, 2006.

[19] L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

[20] J.M. Valério de Carvalho. Exact solution of cutting stock problems using column generation and branch-and-bound. *International Transactions in Operational Research*, 5 :35–44, 1998.

[21] Agence d'évaluation de la recherche et de l'enseignement supérieur. Critères d'identification des chercheurs et enseignants-chercheurs "produisant en recherche et valorisation", http://www.aeres-evaluation.fr/content/download/12877/180830/file/Criteres_Identification_Ensgts-Chercheurs.pdf, mai 2012.

[22] I. Diaz. A branch-and-cut algorithm for graph coloring. *Discrete Applied Mathematics*, 154 :826–847, 2006.

[23] G. Dirac. Some theorems on abstract graphs. *Proceedings of the London Mathematical Society*, 3 :69–81, 1952.

[24] J. Elmirghani and H. Mouftah. All-optical wavelength conversion technologies and applications in DWDM networks. *IEEE Communications Magazine*, 38 :86–92, 2000.

[25] B. Enacheanu, M.C. Alvarez, B. Raison, R. Caire, W. Bienia, O. Devaux, and N. HadjSaid. Optimal meshed distribution network configuration. *International Review of Electrical Engineering*, 4 :957–966, 2008.

[26] L. Fernandes and L. Gouveia. Minimal spanning trees with a constraint on the number of leaves. *European Journal of Operational Research*, 104 :250–261, 1998.

[27] G.R. Filho and L.A.N. Lorena. Constructive genetic algorithm and column generation : An application to graph coloring. In *Proceedings of the 5th Conference of the Association of Asia-Pacific Operations Research Societies within IFORS*, pages 1–8, Singapore, 2000.

[28] République Française. Article 5 de la loi numéro 2003-775 du 21 août 2003 portant réforme des retraites, http://www.legifrance.gouv.fr/affichTexte.do;jsessionid=25317062E7B3685341EBFF1835BD8B86.tpdjo12v_3?cidTexte=LEGITEXT000005635050&dateTexte=20120226f, août 2003.

[29] V. Garcia and P. França. Multiobjective service restoration in electric distribution networks using a local search based heuristic. *European Journal of Operational Research*, 189 :694–705, 2007.

[30] M. Garey and D. Johnson. *Computers and Intractability : a Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[31] L. Gargano, P. Hell P, L. Stacho, and U. Vaccaro. Spanning trees with bounded number of branch vertices. *Lecture Notes in Computer Science*, 2380 :355–363, 2002.

[32] P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting-stock problem. *Operations Research*, 9 :849–859, 1961.

[33] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, 1997.

[34] R. Graham, E. Lawler, J. Lenstra, and A. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling : a survey. *Annals of Discrete Mathematics*, 4 :287–326, 1979.

[35] D. Guthleben. Saviez-vous que . . . ? la revue pour l'histoire du CNRS, http://histoire-cnrs.revues.org/9069, octobre 2009.

[36] R. Hamming. Transcription of the bell communications research colloquium seminar, http://www.cs.virginia.edu/~robins/YouAndYourResearch.pdf, mars 1986.

[37] C. De La Higuera. Networking, http://pagesperso.lina.univ-nantes.fr/~cdlh/Downloads/Networking_2011.pdf, mai 2011.

[38] A. Jajszczyk. Optical networks − the electro-optic reality. *Optical Switching and Networking*, 1 :3–18, 2005.

[39] S. Jasebi, S. Jasebi, and M. Rashidinejad. Application of a novel real genetic algorithm to accelerate the distribution network configuration. *International Review of Electrical Engineering*, 1 :114–121, 2009.

[40] J. Jia, J. Chen, G. Chang, Y. Wen, and J. Song. Multi-objective optimization for coverage control in wireless sensor network with adjustable sensing radius. *Computers & Mathematics with Applications*, 57 :1767–1775, 2009.

[41] J. Kleinberg and A. Kumar. Wavelength conversion in optical networks. *Journal of Algorithms*, 38 :25–50, 2001.

[42] K. Krishnan. *Graph algorithms for loss minimisation through feeder reconfiguration*. PhD thesis, Department of Electrical Engineering, Indian Institute of Science, Bangalore, India, 1998.

[43] C. Lee and H. Kim. Reliable overlay multicast trees for private internet broadcasting with multiple sessions. *Computers & Operations Research*, 34 :2849–2864, 2007.

[44] V. Leggieri, P. Nobili, and C. Triki. Minimum power multicasting problem in wireless networks. *Mathematical Methods of Operations Research*, 68 :295–311, 2008.

[45] M.E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53 :1007–1023, 2005.

[46] R. Montemanni and L.M. Gambardella. Exact algorithms for the minimum power symmetric connectivity problem in wireless networks. *Computers & Operations Research*, 32 :2891–2904, 2005.

[47] O. Mousavi, G. Gharehpetian, and M. Naderi. Estimating risk of cascading blackout using probabilistic methods. In *International Conference on Electric Power and Energy Conversion Systems*, pages 1–4, 2009.

[48] D. Naadef. Polyhedral theory and branch-and-cut algorithm for the symmetric TSP. In G. Gutin and A. Punnen, editors, *The traveling salesman problem and its variations*, pages 29–116. Kluwer Academic Publishers, 2004.

[49] K. Nara and Y. Song. Modern heuristics application to distribution system optimization. In *IEEE Conference Proceedings of the IEEE Power Engineering Society Winter Meeting*, volume 2, pages 826–832, 2002.

[50] G. Oettinger. Quarterly report on european electricity markets. market observatory for energy, `http://ec.europa.eu/energy/observatory/electricity/doc/qreem_\%2009_quarter4.pdf`, 2009.

[51] M. Pinedo. *Planing and Scheduling in Manufacturing and Services*. Springer, New York, 2005.

[52] J. Puchinger and G.R. Raidl. An evolutionary algorithm for column generation in integer programming : An effective approach for 2D bin packing. *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature, Lecture Notes in Computer Science*, 3242 :642–651, 2004.

[53] J. Puchinger and G.R. Raidl. Models and algorithms for three-stage two-dimensional bin packing. *European Journal of Operational Research*, 183 :1304–1327, 2007.

[54] V. Raghunathan, C. Schurgers, S. Park, and M.B. Srivastava. Energy aware wireless microsensor networks. *IEEE Signal Processing Magazine*, 19(2) :1007–1023, 2002.

[55] G.R. Raidl and J. Puchinger. Combining (integer) linear programming techniques and metaheuristics for combinatorial optimization. In *Proceedings of Hybrid Metaheuristics*, pages 31–62, Malaga, 2008.

[56] G. Regulavasa, S. Annapantula, V. Sadhu, and H.R. Bhanagarapu. A new algorithm for minimum loss reconfiguration of distribution networks determining critical switches. *International Review of Electrical Engineering*, 3 :248–252, 2008.

[57] K. Römer and F. Mattern. The design space of wireless sensor networks. *IEEE Wireless Communications*, 11 :54–61, 2004.

[58] A. Rossi, A. Aubry, and M. Jacomino. Connectivity-and-hop-constrained design of electricity distribution networks. *European Journal of Operational Research*, 218 :48–57, 2012.

[59] A. Rossi, M. Sevaux, A. Singh, and M.J. Geiger. On the cover scheduling problem in wireless sensor networks. In *Lecture Notes in Computer Science (LNCS 6701), Network Optimization : International Network Optimization Conference*, pages 657–668, Hamburg, 2011.

[60] A. Rossi, A. Singh, and M. Sevaux. A column generation algorithm for sensor coverage scheduling under bandwidth constraints. *Networks*, To appear.

[61] B. Roy and D. Vanderpooten. An overview on "the european school of mcda : Emergence, basic features and current works". *European Journal of Operational Research*, 99, 1997.

[62] G. Salamon and G. Wiener. On finding spanning trees with few leaves. *Information Processing Letters*, 105 :164–169, 2008.

[63] M. Sevaux. *Métaheuristiques : stratégies pour l'optimisation de la production de biens et de services*. PhD thesis, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes, France, Juillet 2004.

[64] S. Sundar, A. Singh, and A. Rossi. New heuristics for two bounded-degree spanning tree problems. *Information Sciences*, 195 :226–240, 2012.

[65] V. T'Kindt and J-C. Billaut. *Multicriteria scheduling theory, models and algorithms*. Springer, Berlin, 2002.

[66] W. Tomlinson and C. Lin. Optical wavelength-division multiplexer for the 1-1.4-micron spectral region. *Electronics Letters*, 14 :345–347, 1978.

[67] F. Vanderbeck and M. Savelsbergh. A generic view of Dantzig-Wolfe decomposition in mixed integer programming. *Operations research letters*, 34 :296–306, 2006.

[68] L. Volkmann. Estimation for the number of cycles in a graph. *Periodica Mathematica Hungarica*, 33 :153–161, 1996.

[69] C. Wang, M. T. Thai, Y. Li, F. Wang, and W. Wu. Optimization scheme for sensor coverage scheduling with bandwidth constraints. *Optimization letters*, 3(1) :63–75, 2009.

[70] G. Wilfong and P. Winkler. Ring routing and wavelength transmission. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 333–341, 1998.

[71] L. Wolsey. *Integer Programming*. John Wiley and Son, New York, 1998.

[72] FICO. Xpress-MP. `http://www.dashoptimization.com/`, 2009.

[73] J. Yick, B. Mukherjee, and D. Ghosal. Wireless sensor network survey. *Computer Networks*, 52 :2292–2330, 2008.

[74] Z. Zhou, S. Das, and H. Gupta. Variable radii connected sensor cover in sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 5(1) :1–36, 2009.

Chapitre 6

Annexe

6.1 Article à paraitre dans Computers & Operations Research

6.2 Article paru dans Annals of Operations Research

6.3 Article paru dans Discrete Applied Mathematics

Computers & Operations Research ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges

André Rossi [a,*], Alok Singh [b], Marc Sevaux [a]

[a] Lab-STICC, Université de Bretagne Sud – F-56321 Lorient, France
[b] Department of Computer and Information Sciences, University of Hyderabad, Hyderabad 500 046, Andhra Pradesh, India

ARTICLE INFO

Keywords:
Wireless sensor networks
Network lifetime
Column generation
Genetic algorithm

ABSTRACT

This paper addresses the problem of target coverage for wireless sensor networks, where the sensing range of sensors can vary, thereby saving energy when only close targets need to be monitored. Two versions of this problem are addressed. In the first version, sensing ranges are supposed to be continuously adjustable (up to the maximum sensing range). In the second version, sensing ranges have to be chosen among a set of predefined values common to all sensors. An exact approach based on a column generation algorithm is proposed for solving these problems. The use of a genetic algorithm within the column generation scheme significantly decreases computation time, which results in an efficient exact approach.

1. Introduction

Advances in signal processing and embedded systems are at the origin of the growing popularity of Wireless Sensor Networks (WSN) in a wide range of applications [1]. While they were initially used in remote or hostile environments (for battlefield surveillance, or tsunami monitoring), WSN are also increasingly used for health care [12]. Although these applications rely on very different types of sensors, most of them share the following characteristics: sensors operate on a battery that cannot be recharged and a large number of sensors is deployed for improving fault tolerance and lifetime. This paper is concerned with lifetime maximization, that is achieved by making the best possible use of sensors redundancy. Moreover, the sensors are supposed to have adjustable sensing ranges. Such a feature saves energy in the situation where a sensor needs to cover close targets only, as power requirement is a non-decreasing function of the distance between the sensor and the farthest target it covers.

More formally, suppose n sensors are randomly deployed in order to cover a set of m targets $\{\tau_1,\ldots,\tau_k,\ldots,\tau_m\}$. Each sensor s_i has an initial energy b_i. Sensors can either be active or inactive. An inactive sensor does not cover any target, and its power consumption is negligible. When a sensor is active, its power consumption depends on its sensing range. All the sensors have the same maximum sensing range, denoted by R^{max}: a sensor can cover a target if its distance is less than or equal to R^{max}. Lifetime maximization is reached by gathering sensors into non-disjoint subsets called *covers* (each cover being such that each target can be covered by at least one sensor in that cover), and by scheduling these covers, *i.e.*, by determining the amount of time during which each cover is used. The sensors that are not part of the cover that is currently being used are not active. Moreover, covers are not necessarily disjoint, as this allows for reaching longer lifetimes [15]. The schedule must be such that the total amount of energy consumed by sensor s_i is at most equal to its initial energy b_i. The network lifetime is the sum of these durations: when it is exceeded, the coverage of all targets is no longer possible.

This paper addresses two close versions of the lifetime maximization problem:

- Lifetime Maximization with Ad-hoc Sensing Ranges (LM-ASR): Each sensor can adjust its sensing range so as to cover targets with the minimum amount of necessary power (for all targets which distance to the sensor is less than R^{max}). In this model, continuous variations of the sensing range are allowed, this problem version is addressed in [6],
- Lifetime Maximization with Predefined Sensing Ranges (LM-PSR): Sensors have M_{PSR} nonzero and distinct predefined sensing ranges, so all the targets that are under such a predefined range are covered at a predefined power level. In this model, M_{PSR} predefined sensing ranges are supposed to be given, this problem version is addressed in [3].

A mathematical formulation based on integer or linear programming is proposed in [3,6], but is never used for solving the

* Corresponding author. Tel.: +33 297874564; fax: + 33 297874527.
E-mail addresses: andre.rossi@univ-ubs.fr (A. Rossi), marc.sevaux@univ-ubs.fr (M. Sevaux).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.04.001

problem, as both formulations rely on an exponential number of variables. In [17,18,16,11], heuristic methods are proposed for finding the best possible cover for LM-PSR. More precisely, [11] uses a NSGA-II approach, where a trade off is sought between the coverage rate of the sensors in the cover (breach is allowed), the financial cost of the cover (which is proportional to the number of sensors in the cover) and the total power of the sensors used in the cover. These approaches, however, are concerned with the generation of a single cover, and the problem of maximizing lifetime is not addressed.

The present paper generalizes both problems, and provides exact approaches hybridized with metaheuristics for both of them. These approaches are based on column generation, which has been successfully used to address lifetime maximization problems in the literature [2,8,9,14].

The remainder of this paper is organized as follows. Section 2 describes in detail the terminology and notations used in this paper and illustrates them, wherever appropriate, with suitable examples. Section 3 describes the problem models and their resolution approaches. Computational results along with their analysis are presented in Section 4. Finally, Section 5 outlines some concluding remarks and ideas for future works.

2. Definitions and notations

All the notations introduced in this section can be found in Table 1.

2.1. Power consumption and sensor coverage

Dealing with adjustable sensing ranges requires to extend the notion of covers as follows. A cover S_j is a n-column vector of nonnegative reals where $S_{i,j} > 0$ if and only if sensor s_i is part of the cover. $S_{i,j}$ is the power (*i.e.*, the energy consumption rate) of sensor s_i in the cover. If sensor s_i is not part of S_j, then $S_{i,j} = 0$ as this sensor does not consume energy when S_j is used. The amount of energy consumed by active cover S_j is proportional to t_j, where t_j is the amount of time during which cover S_j is used.

The battery of sensor s_i has initial energy b_i and the power consumption of s_i is denoted by $p_i(t)$, as it can vary over time if the sensing range of s_i does. Consequently, the constraint on the limited amount of energy available for sensor s_i can be stated as

$$\int_0^{+\infty} p_i(t)\,dt \le b_i \quad \forall i \in \{1,\ldots,n\} \tag{1}$$

It is assumed that $p_i(t)$ is a function of the sensing range $r(t)$: $p_i(t) = f(r(t))$. As in [3,6], two power consumption models are considered. The first one is referred to as the linear model, the second one is the quadratic model.

$$f_{lin}(r) = p^{max}\left(\frac{r}{R^{max}}\right), \quad 0 \le r \le R^{max}$$

$$f_{qua}(r) = p^{max}\left(\frac{r}{R^{max}}\right)^2, \quad 0 \le r \le R^{max}$$

In both cases, $p_i(t)$ is equal to p^{max} when the sensing range is equal to R^{max} (its maximum value). Since the models proposed in this paper do not depend on the form of the power consumption function, it is denoted by $f(r)$, and is either $f_{lin}(r)$ or $f_{qua}(r)$.

In both LM-ASR and LM-PSR, $r(t)$ can only assume a finite set of numerical values. This is obvious for LM-PSR, and in LM-ASR, $r(t)$ assumes at most as many different values as there are targets under a sensor range. Thus, it can be deduced that $p_i(t)$ is a step function of time. Indeed, the step values are either set by the distance to neighboring targets in LM-ASR, or are taken in a predefined collection of values in LM-PSR. Eq. (1) can then be written as

$$\sum_{j=1}^{c} S_{i,j} t_j \le b_i, \quad \forall i \in \{1,\ldots,n\} \tag{2}$$

where c is the number of covers, and $S_{i,j}$ is the power consumption of sensor s_i associated with the sensing range selected for that cover.

Every sensor s_i is associated with an ordered set D_i containing the targets that it can cover, sorted by increasing power requirement. In an ideal environment, the targets in D_i are those with distance to sensor s_i less than or equal to R^{max}. We assume such an environment, even though this hypothesis is not necessary for the solution approaches proposed in this paper to be valid. The presence of obstacles in the environment, for example, may cause the sensors power consumption not to be a function of $r(t)$ alone as in f. Such a situation can be handled with no inconvenience using one of the two following approaches. If a more realistic function for computing power requirement is available, then it is used instead of f_{lin} of f_{qua}. Otherwise, the network can go through an initialization phase during which the sensing range increases progressively (in LM-ASR), or takes its predefined values sequentially (in LM-PSR), and the targets discovered during that phase are stored along with the corresponding power they require. However in that case, the problem name should refer to adjustable power levels, rather than adjustable sensing ranges.

Whatever the method for computing or measuring power consumption, the targets in D_i are sorted by increasing power requirement, *i.e.*, $D_i(1)$ is the target that can be covered by sensor

Table 1
Notations.

Notation	Meaning
n	Number of sensors
m	Number of targets
s_i	A sensor, for all $i \in \{1,\ldots,n\}$
τ_k	A target, for all $k \in \{1,\ldots,m\}$
b_i	Initial energy of sensor s_i for all $i \in \{1,\ldots,n\}$
$p_i(t)$	Power consumption of sensor s_i over time
D_i	Targets covered by sensor s_i, sorted by increasing power requirement
$p_{i,r}$	Minimum power consumption of sensor s_i required for covering target $D_i(r)$, for all $r \in \{1,\ldots,\lvert D_i\rvert\}$
R^{max}	Sensors' maximum sensing range
M	Maximum number of different power consumption values that a sensor can have
R	Set of the predefined sensing ranges sorted by increasing order (for LM-PSR only)
$C_{k,i}$	Power consumption minimum rank at which sensor s_i can cover target τ_k
S_j	jth cover (*i.e.*, set of sensors). $S_{i,j}$ is the power of sensor s_i in the cover, $\forall j \in \{1,\ldots,c\}$
t_j	Amount of time during which cover S_j is used
$x_{i,r,j}$	Binary decision variable that is set to one iff sensor s_i is part of cover S_j and is used with power $p_{i,r}$

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

s_i with minimum power consumption, whereas $D_i(|D_i|)$ is the target that can be covered by sensor s_i with maximum power consumption. $p_{i,r}$ is defined as the minimum power consumption of sensor s_i required for covering targets $D_i(r)$, for all r in $\{1,\ldots,|D_i|\}$. More formally, $p_{i,r-1} \le p_{i,r}$ for all i in $\{1,\ldots,n\}$, and for all r in $\{2,\ldots,|D_i|\}$. The maximum number of different power consumption values that a sensor can have is denoted by M. In LM-ASR, $M = \max_{i \in \{1,\ldots,n\}} |D_i|$. In LM-PSR, $M = M_{PSR}$. In the case of LM-PSR, R is the set of the predefined sensing ranges sorted by increasing order, with $R_{M_{PSR}} = R^{max}$ and $|R| = M_{PSR}$.

For all $k \in \{1,\ldots,m\}$, $C_{k,i}$ is defined as the power consumption minimum rank at which sensor s_i can cover target τ_k, if the distance between s_i and τ_k is less than or equal to R^{max}. If the distance is greater than R^{max}, then $C_{k,i}$ is set to $M+1$. For example, $C_{k,i} = z$ with $z \in \{1,\ldots,M\}$ means that target τ_k can be covered by sensor s_i, provided that its power is at least $p_{i,z}$. If $z = M+1$, then target τ_k is out of range of s_i.

2.1.1. Example

For the sake of illustration, the ordered sets D, C and p are computed for the two problem versions LM-ASR and LM-PSR, with the one-dimensional example shown in Fig. 1. The horizontal axis allows the reader to assess the distances more easily.

There are $n=2$ sensors, $m=3$ targets, the maximum sensing range R^{max} is set to 4 and sensors power requirements are given by $f_{qua}(r)$ with $p^{max} = 1$. It is also assumed that $b_1 = b_2 = 1$. This implies that each sensor can cover all the targets which are at distance less than 4 from it, for 1 s before its battery is depleted. Naturally, the lifetime can be extended further if sensors reduce their sensing range.

The ordered sets D are common to both LM-ASR and LM-PSR, they are defined as

$$D_1 = (1,2), \quad D_2 = (2,3,1)$$

Indeed, it can be seen from Fig. 1 that sensor s_1 can cover targets τ_1 and τ_2, and τ_1 is closer to s_1 than τ_2. Sensor s_2 can cover all the three targets, namely τ_2, τ_3 and τ_1 when sorted by non decreasing distance to s_2. Equivalently, we may have $D_2 = (3,2,1)$.

Sets C and p are different according to the problem version:

- LM-ASR

  The ordered sets defining the power consumption of sensors are defined below:

  $$p_1 = (\tfrac{1}{4},1), \quad p_2 = (\tfrac{1}{16},\tfrac{1}{16},\tfrac{9}{16})$$

  Sensor s_1 is at distance 2 from the closest target it covers, hence $p_{1,1} = (2/R^{max})^2 = 0.25$. The second target in D_1 is at distance R^{max} from s_1, hence $p_{1,2}$ reaches its maximum value (*i.e.*, 1).
  Sensor s_2 is at distance 1 from the closest target it covers, hence $p_{2,1} = (1/R^{max})^2 = \frac{1}{16}$. The second target in D_2 is also at distance 1 from s_2, hence $p_{2,2} = \frac{1}{16}$, and target τ_1 is at distance 3 so $p_{2,3} = \frac{9}{16}$.
  The maximum cardinality of D_i over i in $\{1,\ldots,n\}$ defines the maximum number of different sensing ranges that a sensor can have. Here, s_1 has two sensing ranges, and s_2 has three, so $M=3$.

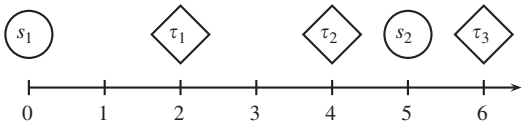


Fig. 1. A one-dimensional example.

  The ordered sets C_k that indicate the rank in power consumption of the sensors that cover target τ_k are

  $$C_1 = (1,3), \quad C_2 = (2,1), \quad C_3 = (4,2)$$

  Target τ_1 is covered by s_1, which can cover it using its first (lower) sensing range. τ_1 is also covered by s_2, but it requires its third sensing range to reach that target.
  Target τ_2 is covered by s_1, which can cover it using its second (highest) sensing range; it is also covered by s_2 but only requires its lower sensing range to reach it.
  Finally, target τ_3 is not covered by s_1, hence $C_{3,1}$ is set to $M+1=4$. Target τ_3 is covered by sensor s_2, and requires its second sensing range (or its first sensing range, as they are equal).

- LM-PSR

  It is supposed that $M_{PSR} = 2$, and the two predefined sensing ranges are $R = \{2,4\}$. By definition of $f_{qua}(r)$, the corresponding power consumption levels are $\frac{1}{4}$ and 1, respectively.
  The ordered sets defining the power consumption of sensors are

  $$p_1 = (\tfrac{1}{4},1), \quad p_2 = (\tfrac{1}{4},\tfrac{1}{4},1)$$

  Sensor s_1 is at distance 2 from the closest target it covers, hence the first predefined sensing range is needed so $p_{1,1} = (\frac{2}{4})^2 = 0.25$. The second target in D_1 is at distance R^{max} from s_1, hence $p_{1,2}$ reaches its maximum value (*i.e.*, 1).
  Sensor s_2 is at distance 1 from the closest target it covers, so $p_{2,1} = \frac{1}{4}$. The second target in D_2 is at distance 2 from s_2, hence $p_{2,2} = \frac{1}{4}$: both targets require the first predefined sensing range to be used. Target τ_1 is at distance 3 so the second predefined sensing range is used, leading to $p_{2,3} = 1$.
  The maximum number of different sensing ranges is equal to the number of predefined sensing ranges so $M=2$.
  The ordered sets C_k that indicate the rank in power consumption of the sensors that cover target τ_k are

  $$C_1 = (1,2), \quad C_2 = (2,1), \quad C_3 = (3,1)$$

  Target τ_1 is covered by s_1, which can cover it using the first predefined sensing range. τ_1 is also covered by s_2, but it requires the second predefined sensing range to reach that target.
  Target τ_2 is covered by s_1, which can cover it using the second sensing range; it is also covered by s_2 which can cover it using the first sensing range to reach it.
  Finally, target τ_3 is not covered by s_1, hence $C_{3,1}$ is set to $M+1=3$. Target τ_3 is covered by sensor s_2, and requires the first predefined sensing range.

2.2. Cover dominance properties

This section introduces two equivalent ways of representing covers. Those two different encodings are necessary as they are both used in the column generation approach. Then, theoretical properties are provided for characterizing non-dominated covers that are useful for maximizing lifetime.

2.2.1. Cover encodings

Real vector encoding: Cover S_j can be defined as a n-column real vector where $S_{i,j}$ is the power consumption of sensor s_i, *i.e.*, for all $i \in \{1,\ldots,n\}$, $S_{i,j}$ belongs to the discrete set $\{0,p_{i,1},p_{i,2},\ldots,p_{i,|D_i|}\}$ for LM-ASR, and $S_{i,j}$ belongs to the discrete set $\{0,p_{i,1},p_{i,2},\ldots,p_{i,M_{PSR}}\}$ for LM-PSR. For both problems, $S_{i,j} = 0$ if and only if s_i is not part of cover S_j.

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

Binary encoding: Equivalently, S_j can be represented as a set of n ordered lists of $|D_i|$ binary numbers for LM-ASR, and M_{PSR} binary numbers for LM-PSR. More precisely, $x_{i,r,j}$ is set to one if sensor s_i is part of the cover and is used with power $p_{i,r}$, for all $i \in \{1,\ldots,n\}$, it is zero otherwise. In addition, at most one variable $x_{i,r,j}$ should be set to one for all $i \in \{1,\ldots,n\}$ as each sensor has at most one power consumption level (and no level at all if it is not part of the cover).

Changing encoding: The following equation is used for transforming a cover from the binary encoding to the real vector encoding in LM-ASR:

$$S_{i,j} = \sum_{r=1}^{|D_i|} x_{i,r,j} p_{i,r} \quad \forall i \in \{1,\ldots,n\}, \quad \forall j \in \{1,\ldots,c\} \tag{3}$$

For LM-PSR, $|D_i|$ and $p_{i,r}$ must be replaced with M_{PSR} and $f(R_r)$, respectively.

The reciprocal transformation is given in Algorithm 1 for LM-ASR.

Algorithm 1. From the real vector encoding to the binary encoding.

```
for i=1 to n do
    for r=1 to |D_i| do
        if S_{i,j} = p_{i,r} then
            x_{i,r,j} ← 1;
        else
            x_{i,r,j} ← 0;
```

Algorithm 1 can be used for LM-PSR, by replacing $|D_i|$ and $p_{i,r}$ with M_{PSR} and $f(R_r)$, respectively.

2.2.2. Non-dominated covers

All the approaches that use column generation for addressing lifetime maximization in wireless sensor networks rely on the notion of cover [2,9,14]. In all these column generation approaches, the main issue is to find a cover that is likely to increase further the lifetime of the current solution. To the best of our knowledge, this article provides the first theoretical study on how to reduce the search space of such attractive covers. As a result, it is shown that the search space can be reduced to non-dominated covers. This provides a significant advantage over performing a search in the much larger set of valid covers.

A cover is said to be *valid* if the power level of its sensors is such that all the targets are covered by at least one sensor. A valid cover S_j is said to be *dominated* if there exists another valid cover $S_{j'}$ containing a subset of sensors of S_j that are used at a lower power level, and at least one sensor is used in $S_{j'}$ at a strictly lower power than in S_j. More formally, S_j is dominated if there exists another valid cover $S_{j'}$ such that $S_{i,j'} \leq S_{i,j}$ for all $i \in \{1,\ldots,n\}$, and there exists at least one integer i_0 in $\{1,\ldots,n\}$ such that $S_{i_0,j'} < S_{i_0,j}$.

Therefore, in any non-dominated cover, decreasing the power level of any sensor compromises coverage. Consequently, a non-dominated cover can be built from any valid cover by decreasing the power of sensors as long as it is possible to do so without compromising targets coverage. This process is referred to as cover refinement and is implemented in Algorithm 2.

Lemma 1. *There exists an optimal solution to LM-ASR (or LM-PSR) in which non-dominated covers only are used a non-zero amount of time.*

Proof. Lemma 1 is proved by contradiction. Suppose that all optimal solutions to LM-ASR (or LM-PSR) are such that there exists a dominated cover S_j used for a nonzero amount of time. Then a non-dominated cover $S_{j'}$ can be built from S_j, and be used instead of S_j in the optimal solution. Doing so preserves target coverage by definition of non-dominated covers. Repeating this process for all dominated covers used a nonzero amount of time leads to an optimal solution to LM-ASR (or LM-PSR) where non-dominated covers only are used a non-zero amount of time. Contradiction.

As a consequence of Lemma 1, the search for an optimal solution to LM-ASR (or LM-PSR) can be restricted to non-dominated covers only. Enforcing this restriction is efficient when designing a solution approach to LM-ASR or LM-PSR, as non-dominated covers are generally a small fraction of valid covers.

Lemma 2. *The number of sensors in a non-dominated cover is at most m.*

Proof. Each target has to be covered by at least one sensor, so at most m sensors are required for covering all the targets. If a valid cover has more than m sensors, then one of them can be removed, so it is dominated.

Lemma 2 can be seen as a necessary (but not sufficient) condition for a cover to be non-dominated. It is used in the column generation algorithm for building candidate non-dominated covers in Section 3.3.

2.2.3. Example

The Example introduced in Section 2.1.1 is used again for illustrating both cover encodings, valid covers, dominated and non-dominated covers, for LM-ASR and LM-PSR.

- LM-ASR
  There exist 5 valid covers, represented under the real vector encoding:

$$S_1^{ASR} = \begin{bmatrix} \frac{1}{4} \\ \frac{1}{16} \end{bmatrix}, S_2^{ASR} = \begin{bmatrix} 0 \\ \frac{9}{16} \end{bmatrix}, S_3^{ASR} = \begin{bmatrix} 1 \\ \frac{1}{16} \end{bmatrix}, S_4^{ASR} = \begin{bmatrix} 1 \\ \frac{9}{16} \end{bmatrix}, S_5^{ASR} = \begin{bmatrix} \frac{1}{4} \\ \frac{9}{16} \end{bmatrix}$$

  Covers S_3^{ASR}, S_4^{ASR} and S_5^{ASR} are dominated because of cover S_1^{ASR}. Hence, S_1^{ASR} and S_2^{ASR} are non-dominated.
  The binary encoding of covers S_1^{ASR} and S_2^{ASR} are X_1^{ASR} and X_2^{ASR}, respectively:

$$X_1^{ASR} = \begin{bmatrix} 1 & 0 & - \\ 1 & 0 & 0 \end{bmatrix}, X_2^{ASR} = \begin{bmatrix} 0 & 0 & - \\ 0 & 0 & 1 \end{bmatrix}$$

  The – sign on the first row indicates that s_1 has only two power consumption levels ($|D_1| = 2$). In S_1^{ASR}, both sensors are part of the cover, and are used with their minimum power. In S_2^{ASR}, only s_2 is part of the cover, and is used at its maximum power.
- LM-PSR
  There are also five valid covers for LM-PSR, they are shown below under the real vector encoding:

$$S_1^{PSR} = \begin{bmatrix} \frac{1}{4} \\ \frac{1}{4} \end{bmatrix}, S_2^{PSR} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, S_3^{PSR} = \begin{bmatrix} 1 \\ \frac{1}{4} \end{bmatrix}, S_4^{PSR} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, S_5^{PSR} = \begin{bmatrix} \frac{1}{4} \\ 1 \end{bmatrix}$$

  Again, S_3^{PSR}, S_4^{PSR}, and S_5^{PSR} are dominated. The binary encoding for S_1^{PSR} and S_2^{PSR} are X_1^{PSR} and X_2^{PSR}, respectively:

$$X_1^{PSR} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}, X_2^{PSR} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

  Unlike LM-ASR, all the rows in the binary encoding of a cover have the same size in LM-PSR because each sensor has exactly M_{PSR} power consumption levels.

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

3. Problem modeling

3.1. A mixed integer linear programming model

LM-ASR can be modeled as the following mathematical formulation:

$$\text{Max} \quad \sum_{j=1}^{c} t_j \tag{4}$$

$$\text{s.t.} \quad \sum_{j=1}^{c} \left(\sum_{r=1}^{|D_i|} x_{i,r,j} p_{i,r} \right) t_j \le b_i, \quad \forall i \in \{1,\ldots,n\} \tag{5}$$

$$\sum_{r=1}^{|D_i|} x_{i,r,j} \le 1 \quad \forall i \in \{1,\ldots,n\}, \forall j \in \{1,\ldots,c\} \tag{6}$$

$$\sum_{i=1}^{n} \sum_{r=C_{k,i}}^{|D_i|} x_{i,r,j} \ge 1 \quad \forall k \in \{1,\ldots,m\}, \ \forall j \in \{1,\ldots,c\} \tag{7}$$

$$\sum_{i=1}^{n} \sum_{r=1}^{|D_i|} x_{i,r,j} \le m \quad \forall j \in \{1,\ldots,c\} \tag{8}$$

$$x_{i,r,j} \in \{0,1\} \quad \forall i \in \{1,\ldots,n\}, \ \forall r \in \{1,\ldots,|D_i|\}, \ \forall j \in \{1,\ldots,c\} \tag{9}$$

$$t_j \ge 0 \quad \forall j \in \{1,\ldots,c\} \tag{10}$$

The objective function is to maximize the network lifetime. Eq. (5), which is non-linear, ensures that energy limitations are respected for each sensor. It is a combination of Eqs. (2) and (3). Eq. (6) states that in each cover, each sensor is used with at most one power level. The target coverage requirement are enforced by Eq. (7), Eq. and (8) restrict the search for covers with at most m sensors. This model can be used for LM-PSR by replacing $|D_i|$ and $p_{i,r}$ with M_{PSR} and $f(R_r)$, respectively.

This mathematical model is linearized by introducing continuous variables $z_{i,r,j}$ for replacing $x_{i,r,j} p_{i,r} t_j$ in Eq. (5). These new variables are linked with $x_{i,r,j}$ and t_j as follows:

$$z_{i,r,j} \le p_{i,r} t_j \quad \forall i \in \{1,\ldots,n\}, \ \forall r \in \{1,\ldots,|D_i|\}, \ \forall j \in \{1,\ldots,c\} \tag{11}$$

$$z_{i,r,j} \le b_i x_{i,r,j} \quad \forall i \in \{1,\ldots,n\}, \ \forall r \in \{1,\ldots,|D_i|\}, \ \forall j \in \{1,\ldots,c\} \tag{12}$$

$$z_{i,r,j} \ge p_{i,r} t_j + b_i (x_{i,r,j} - 1), \quad \forall i \in \{1,\ldots,n\}, \ \forall r \in \{1,\ldots,|D_i|\}, \ \forall j \in \{1,\ldots,c\} \tag{13}$$

$$z_{i,r,j} \ge 0 \quad \forall i \in \{1,\ldots,n\}, \ \forall r \in \{1,\ldots,|D_i|\}, \ \forall j \in \{1,\ldots,c\} \tag{14}$$

As the objective function will push $z_{i,r,j}$ downward, Eqs. (11) and (12) are useless. The linearized model is then

$$\text{Max} \quad \sum_{j=1}^{c} t_j$$

$$\text{s.t.} \quad \sum_{j=1}^{c} \sum_{r=1}^{|D_i|} z_{i,r,j} \le b_i \quad \forall i \in \{1,\ldots,n\}$$

$$z_{i,r,j} \ge p_{i,r} t_j + b_i (x_{i,r,j} - 1) \quad \forall i \in \{1,\ldots,n\}, \ \forall r \in \{1,\ldots,|D_i|\}, \ \forall j \in \{1,\ldots,c\}$$

$$\sum_{r=1}^{|D_i|} x_{i,r,j} \le 1 \quad \forall i \in \{1,\ldots,n\}, \ \forall j \in \{1,\ldots,c\}$$

$$\sum_{i=1}^{n} \sum_{r=C_{k,i}}^{|D_i|} x_{i,r,j} \ge 1 \quad \forall k \in \{1,\ldots,m\}, \ \forall j \in \{1,\ldots,c\}$$

$$\sum_{i=1}^{n} \sum_{r=1}^{|D_i|} x_{i,r,j} \le m \quad \forall j \in \{1,\ldots,c\}$$

$$x_{i,r,j} \in \{0,1\} \quad \forall i \in \{1,\ldots,n\}, \ \forall r \in \{1,\ldots,|D_i|\}, \ \forall j \in \{1,\ldots,c\}$$

$$t_j \ge 0 \quad \forall j \in \{1,\ldots,c\}$$

$$z_{i,r,j} \ge 0 \quad \forall i \in \{1,\ldots,n\}, \ \forall r \in \{1,\ldots,|D_i|\}, \ \forall j \in \{1,\ldots,c\}$$

Table 2
Using the MILP formulation of LM-ASR on four instances.

Instances	n	m	Best sol.	Opt. sol.
n012m005	12	5	4.0459	4.0459
n025m010	25	10	4.8780	4.8780
n050m015	50	15	11.6013	14.0702
n100m030	100	30	19.8385	38.9922

This mixed integer linear problem (MILP) has a very large number of variables and constraints, and it may be used for addressing very small instances only. Indeed, we have solved that formulation for LM-ASR with the commercial solver Xpress-MP [7] on the four instances whose characteristics are shown in Table 2 (the hardware and software used in all numerical experiments is described in Section 4). The last column displays the optimal objective value returned by the column generation algorithm. The solver was allowed to solve the MILP formulation for 1 h on each instance, and none of them could be solved to optimality: even if the optimal solution was found for the first two instances, the solver could not prove them optimal. Moreover, it can be seen that solution quality drastically decreases when problem size increases.

The last two instances of Table 2 are also used in Section 4, and can be solved to optimality in less than 10 s with the column generation based algorithms introduced in Section 3.2. In addition, as solving the LP relaxation of the MILP formulation requires even more time than solving the problem to optimality with the column generation based algorithms, using rounding procedures for generating initial covers for the column generation algorithms is too costly. However, this MILP formulation shows that variables t_j and $x_{i,r,j}$ are connected together with a single constraint, which suggests to perform a Dantzig–Wolfe decomposition, resulting in the proposed column generation algorithm.

3.2. Master problem

LM-ASR and LM-PSR can be formulated as the following linear program, where S_j is a non-dominated cover written under the real vector encoding, and c is the number of non-dominated covers:

$$\text{Max} \quad \sum_{j=1}^{c} t_j$$

$$\text{s.t.} \quad \sum_{j=1}^{c} S_{i,j} t_j \le b_i \quad \forall i \in \{1,\ldots,n\}$$

$$t_j \ge 0 \quad \forall j \in \{1,\ldots,c\}$$

The objective function is the sensor network lifetime, and the constraints enforce that each battery has a limited amount of energy. This linear program cannot be solved in practice because it requires enumerating the exhaustive set of non-dominated covers, which cardinality is exponential in n.

Furthermore, as this linear program has n constraints (apart from nonnegativity), a feasible basis is a collection of at most n variables t_j, therefore at most n non-dominated covers are used for a nonzero amount of time in any basic optimal solution [4].

That is the reason why the proposed approach relies on column generation [13]. This iterative approach consists in

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

solving the linear program above with a limited number c of columns (this problem is referred to as the master problem), then the auxiliary problem introduced in Section 3.3 is solved for generating an attractive column, *i.e.*, a non-dominated cover to be introduced in the master problem for further maximizing its objective function. To this end, cover S_j is built so as to maximize the reduced cost that its associated variable t_j would have once introduced in the master problem. If its reduced cost is strictly positive, S_j is actually added to the master problem as a new column, and the master problem is solved again. This iterative process stops when it is no longer possible to find a non-dominated cover with a strictly positive reduced cost: this proves that the current solution to the master problem is optimal, and the algorithm stops.

The master problem is initialized with a single cover (*i.e.*, initially $c=1$) that involves all the sensors, used at their maximum power level. If the master problem is infeasible, then there exists at least one target out of the range of any sensor, hence the problem has no solution. This initial cover is obviously dominated, but replacing it with a non-dominated one has a very marginal impact on convergence. This is due to the fact that this cover becomes useless and is replaced with more efficient ones after a few iterations.

3.3. Auxiliary problem ILP formulation

At every iteration, the master problem finds the optimal lifetime value based on a restricted set of c columns. The auxiliary problem is to generate an additional attractive column, referred to as cover S_{c+1}, that may allow to increase lifetime further once it is introduced in the master problem. If such a column is found, it is added to the master problem, c is increased by one and the master problem is resolved. If no such column exists, then the current solution of the master problem is optimal, as introducing even all the columns would not allow to increase the lifetime.

Unlike the master problem, the cover under construction in the auxiliary problem is written under the binary encoding. As a cover (or a column) is said to be attractive if and only if its reduced cost is strictly positive, the auxiliary problem objective function is to maximize the reduced cost value associated with the cover S_{c+1}, which is $1-y^T S_{c+1}$, where y is the vector of dual variables of the master problem solved in the current iteration. The auxiliary problem for LM-ASR consists of Eqs. (15)–(19). The auxiliary problem for LM-PSR is obtained by replacing every occurrence of $|D_i|$ and $p_{i,r}$ with M_{PSR} and $f(R_r)$, respectively:

$$\text{Max} \quad 1-\sum_{i=1}^{n}\left(\sum_{r=1}^{|D_i|} x_{i,r,c+1}\,p_{i,r}\right)y_i \tag{15}$$

$$\text{s.t.} \quad \sum_{r=1}^{|D_i|} x_{i,r,c+1} \le 1 \quad \forall i \in \{1,\ldots,n\} \tag{16}$$

$$\sum_{i=1}^{n}\sum_{r=C_{k,i}}^{|D_i|} x_{i,r,c+1} \ge 1 \quad \forall k \in \{1,\ldots,m\} \tag{17}$$

$$\sum_{i=1}^{n}\sum_{r=1}^{|D_i|} x_{i,r,c+1} \le m \tag{18}$$

$$x_{i,r,c+1} \in \{0,1\} \quad \forall i \in \{1,\ldots,n\}\ \forall r \in \{1,\ldots,|D_i|\} \tag{19}$$

The decision variables are $x_{i,r,c+1}$, whereas y_i, $p_{i,r}$, C_k and D_i are given data. The objective function is the reduced cost of the new cover, where $S_{i,c+1}$ have been substituted using Eq. (3). Constraint (16) ensures that each sensor is associated to a single power level (or no power level at all if it is not part of the cover). Constraint (17) states that the cover must be valid, *i.e.*, all targets must be covered by at least one sensor. Note that if sensor s_i does not cover target τ_k, then $r=M+1$ and so no term involving $x_{i,r,c+1}$ is part of the sum. Constraint (18) is the necessary condition stated in Lemma 2 for the cover to be non-dominated. Finally, constraint (19) enforce binary requirements.

The auxiliary problem returns S_{c+1} if its objective value is strictly positive, otherwise the algorithm stops as a zero objective value implies that there does not exists any attractive cover, hence the master problem current solution is optimal.

3.3.1. Relationship between LM-ASR and LM-PSR

The auxiliary problem can be strengthened in the case of LM-PSR. If a sensor is such that there is no target located at distance $d \in]R_r, R_{r+1}]$ with $r \in \{1,\ldots,M_{PSR}-1\}$, then the corresponding power level (namely $f(R_r)$) is never used by this sensor. Consequently, variable $x_{i,r,c+1}$ can be set to zero. Thus, the following equation is added to the auxiliary problem:

$$x_{i,r,c+1} = 0 \quad \forall (i,r) \in \{1,\ldots,n\} \times \{1,\ldots,M_{PSR}\}, \quad f(R_r) \notin p_i \tag{20}$$

Such a situation never happens with LM-ASR, as power levels are defined for each sensor for exactly fitting its neighboring targets with minimum power requirement. Eq. (20) is then specific to LM-PSR.

Despite constraint (18), the cover returned after solving the auxiliary problem may not always be non-dominated. Algorithm 2 is a post-optimization procedure that aims at refining S_{c+1} (under binary encoding) so as to make it non-dominated.

Algorithm 2. Refining S_{c+1} (under binary encoding) into a non-dominated cover.

```
for i = 1 to n do
    for r = |D_i| to 1 do
        if x_{i,r,c+1} = 1 then
            x_{i,r,c+1} ← 0;
            if r > 1 then
                x_{i,r-1,c+1} ← 1;
            breach ← false;
            k ← 1;
            while breach = false and k ≤ m do
                if Σ_{i'=1}^{n} Σ_{r'=C_{k,i'}}^{|D_{i'}|} x_{i',r',c+1} = 0 then
                    breach ← true;
                k ← k + 1;
            if breach = true then
                x_{i,r,c+1} ← 1;
                if r > 1 then
                    x_{i,r-1,c+1} ← 0;
```

Algorithm 2 attempts to reduce the power of all sensors in S_{c+1} to its closest lower level. If the considered sensor is already using its minimum power, it is tested for removal. Then, the coverage constraints (Eq. (17)) are checked. If one of them does not hold, the Boolean variable *breach* is set to true, and the sensor is set back to its original power level. Otherwise, further power reductions and removals are attempted. When this algorithm terminates, the resulting cover is non-dominated as reducing the power or removing any sensor compromises target coverage.

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

Lemma 3. *Whatever M_{PSR} and the values for the predefined sensing ranges, the objective function value reached by LM-ASR is always larger than or equal to the objective function value reached by LM-PSR.*

Proof. Let $\mathcal{S}^{PSR}$ be an optimal solution to LM-PSR consisting of c covers denoted by S_j^{PSR}, each of them being used a nonzero amount of time t_j. A feasible solution to LM-ASR consisting in c covers used the same amount of time and denoted by $\{S_1^{ASR},\ldots,S_c^{ASR}\}$ is built from $\mathcal{S}^{PSR}$ by transforming the cover S_j^{PSR} for all j in {1, …, c} as follows.

For all i in $\{1,\ldots,n\}$, $S_{i,j}^{ASR}$ is set to $p_{i,r}$ where r is the maximum integer such that $p_{i,r} \le S_{i,j}^{PSR}$. Thus, sensor s_i covers the same targets in S_j^{PSR} and S_j^{ASR} because by definition of $p_{i,r}$ the targets covered by s_i are the same for any power value in $[p_{i,r},p_{i,r+1}[$. Consequently, $S_j^{ASR} \le S_j^{PSR}$ for all $j \in \{1,\ldots,c\}$ so S_j^{ASR} can be used for t_j units of time, leading to the same lifetime duration as $\mathcal{S}^{PSR}$. □

Lemma 4. *LM-PSR and LM-ASR have the same optimal objective value if*

$$\bigcup_{i \in \{1,\ldots,n\} r \in \{1,\ldots,|D_i|\}} p_{i,r} \subseteq \bigcup_{r \in \{1,\ldots,M_{PSR}\}} f(R_r) \tag{21}$$

where $p_{i,r}$ refer to power levels in LM-ASR, $f(R_r)$ being the power levels in LM-PSR, associated with predefined sensing ranges.

Proof. If Eq. (21) holds, then any non-dominated cover for LM-ASR is also non-dominated for LM-PSR. Then, LM-PSR is identical to LM-ASR: for each sensor, all useless power levels are associated a zero decision variable in the auxiliary problem by Eq. (20). Once zero decision variables are removed, the auxiliary problems for LM-ASR and LM-PSR are identical. □

The example in Fig. 2 shows that Eq. (21) is not a necessary condition for LM-PSR and LM-ASR to have identical optimal objective values. Indeed, all power levels in LM-ASR may not be used in an optimal solution. More specifically, s_1 has to maintain its sensing range to $R^{max}=4$ for covering target τ_2, hence its sensing range cannot be decreased to 2. In that case, LM-PSR with only one range (R^{max}) and LM-ASR reach the same optimal lifetime (1 s), whereas Eq. (21) does not hold as $p_1=(\frac{1}{4},1)$ and $f(R_1)=1$.

3.3.2. Example

The example introduced in Section 2.1.1 is used for illustrating the proposed approach as well as Lemma 3.

- LM-ASR
  The initial cover is S_4^{ASR}. At the first iteration, the master problem optimal objective value is $LT=1$, and the auxiliary problem generates a second cover, which turns out to be S_2^{ASR}. At the second iteration, the master problem optimal objective value is $LT=\frac{16}{9}\approx 1.777$, with $t_1=1$ and $t_2=\frac{7}{9}$. The auxiliary problem generates a third cover, which is S_1^{ASR}. At the third iteration, the master problem optimal objective value is $LT=\frac{16}{3}\approx 5.333$, with $t_1=0$, $t_2=\frac{4}{3}$ and $t_3=4$. Then, the auxiliary problem has a zero optimal objective value, which proves that the master problem current solution is optimal.

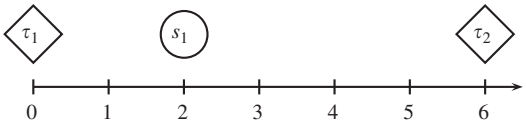


Fig. 2. A one-dimensional example for showing that the converse of Lemma 4 is not true.

- LM-PSR
  The initial cover is S_4^{PSR}. At the first iteration, the master problem optimal objective value is $LT=1$, and the auxiliary problem generates a second cover, which turns out to be S_1^{PSR}. At the second iteration, the master problem optimal objective value is $LT=4$, with $t_1=0$ and $t_2=4$, and the auxiliary problem has a zero optimal objective value, which proves that this solution is optimal.

As shown in Lemma 3 the objective function value of LM-ASR is larger than the one of LM-PSR.

3.4. A genetic algorithm for the auxiliary problem

Solving the auxiliary problem to optimality is sometimes very time-consuming, and results in the production of a single cover. This section describes the main features of the genetic algorithm, referred to as *GA* that we designed for overcoming these drawbacks. More precisely, in the proposed approach, the regular way for generating attractive covers is through *GA*. Upon failure, *GA* is again run afresh one more time, and ILP is used as last resort (*i.e.*, when both runs of *GA* have failed to find any attractive cover), and for proving that the current solution to the master problem is optimal.

Chromosome representation and fitness evaluation: Each chromosome represents a cover and is encoded using the real vector encoding introduced in Section 2.2.1. Three fitness functions are used. The primary fitness function is the objective function of the auxiliary problem that has to be maximized. The secondary fitness function is the power of the cover. The third fitness function is the number of sensors in the cover. The secondary fitness function is needed for distinguishing between two covers having the same value for the primary fitness function. Similarly, the third fitness function is needed for distinguishing between two covers having the same values for primary and secondary fitness functions. A cover is considered to be better than the other, if either it has a higher value according to the primary fitness function or the primary fitness values are equal but it consumes less power than the other or both the primary and secondary fitness values are equal and it has fewer sensors than the other. Three fitness functions are needed because many covers have the same value for the primary and secondary fitness functions. This happens more often in LM-PSR than in LM-ASR.

Selection: Probabilistic binary tournament selection is used to select the two parents for crossover. In probabilistic binary tournament selection, two candidates are picked uniformly at random from the current population and their fitness is compared. With probability π_b, the candidate which has better fitness is selected to be a parent, otherwise the candidate with worse fitness is selected. The second parent is selected in an analogous manner.

Crossover: We have used the two crossover operators. The first crossover operator that is used simply transfers each sensor present in either of the two parents to the child chromosome. However, if a sensor is present in both the parents and have different power levels then for such a sensor the child will have the higher of the two power levels with probability π_h, otherwise it will have the lower of the two power levels. The second crossover that is used is the uniform crossover. The first crossover is used with probability π_f, otherwise the second crossover is used. The two crossovers are needed to maintain a balance between exploitation and exploration.

Mutation: Mutation is applied to the child obtained through crossover. The mutation operator considers each sensor one-by-one, and, if it is present in the child, it deletes it with probability

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

π_m. If a sensor is not present then mutation operator inserts it at a random power level with probability π_m.

Repair operator The child chromosome obtained after the application of genetic operators may not be feasible. Therefore, a repair operator is designed, which not only converts the child into a feasible cover, but also into a good non-dominated cover. It consist of three stages. The first stage transforms the child into a feasible cover. The second stage tries to improve the value of the auxiliary problem's objective function (hereafter, refers to as objective function in the remainder of this section) by removing some sensors. The third stage tries to further improve the value of the objective function by reducing the range of some sensors.

The first stage considers each target one-by-one and if the target under consideration is not covered by any sensor then it either adds a new sensor to the cover or increases the power level of an existing sensor in a way that covers the target in question and that leads to minimum reduction in the value of the objective function. Coverage information of all targets are updated before considering the next target.

The second stage deletes those sensors from the cover all of whose covered targets are covered by other sensors also. It follows an iterative process. During each iteration it begins by computing the set S_{red} of those sensors in the cover, all of whose targets are redundantly covered. Then from this set it removes the sensor s_i corresponding to the maximum $y_i \times p_{i,r}$ value from the cover, where $p_{i,r}$ is the current power level of sensor s_i. This process is repeated, until the iteration, where S_{red} is found to be empty. Note that deleting sensor with maximum $y_i \times p_{i,r}$ value leads to maximum increase in the value of the objective function.

The third stage ensures that remaining sensors are used only at their respective minimum power levels necessary for covering all targets. It follows an iterative process where during each iteration it decreases the power of a sensor from its current level to its next (strictly) lower level if doing so does not result into a coverage breach. If there are more than one candidate sensors, then the power level of sensor with maximum $y_i \times (p_{i,r} - p_{i,q})$ value is reduced from its current value $p_{i,r}$ to its next (strictly) lower level $p_{i,q}$. Note that for LM-PSR, $(r-q)=1$, whereas for LM-ASR as many power levels can be equal so $(r-q) \geq 1$. This process is repeated until it is not possible to reduce the power level of any sensor.

Replacement policy: Our genetic algorithm uses steady-state population replacement method [5]. In this method genetic algorithm repeatedly selects two parents, performs crossover and mutation to generate a single child that replaces a less fit member of the population. This is different from generational method where the entire parent population is replaced with an equal number of newly created children every generation. In comparison to the generational method, the steady-state population replacement method generally finds better solutions faster. This is due to keeping the best solutions in the population permanently and the immediate availability of the generated child for selection and reproduction. Another advantage of the steady-state population replacement method is the ease with which duplicate copies of the same individuals are avoided in the population. In the generational method, duplicate copies of the highly fit individuals may exist in the population. Within few generations, these highly fit individuals can dominate the whole population. When this happens, the crossover becomes totally ineffective and the mutation becomes the only possible way to improve solution quality. When this happens, improvement, if any, in solution quality is quite slow. Such a situation is known as the premature convergence. In the steady-state method, we can easily prevent this situation by simply comparing each newly generated child with current population members and discarding the child, if it is identical to any current member.

Initial population generation: Each member of the initial population is generated randomly by following a three stage procedure. With probability π_a, the first stage includes as many sensors as possible with $y_i=0$ at their maximum power level. Actually, these sensors do not contribute to the objective function irrespective of their power levels. These sensors are added to the solution one-by-one in some random order and the coverage information of all the target are updated whenever a sensor is added to the solution. The remaining uncovered targets are covered by following an iterative approach. During each iteration an uncovered target is selected randomly and then a sensor that can cover this target is also selected randomly. This sensor can be a new sensor or a sensor already present in the cover but with a power level lower than required for covering the target under consideration. The power level of this sensor is set to the minimum power level necessary for covering the target in question. The coverage information of all the targets are updated. This process is repeated until no target is left uncovered. The second and third stages are exactly similar to the second and third stages of repair operator except during each iteration a candidate is selected randomly. This is done to increase the diversity of initial population.

Each newly generated chromosome is checked for uniqueness against the population members generated so far, and, if it is unique, then it is included in the initial population, otherwise it is discarded.

Other features: If our genetic algorithms found an attractive cover then up to *MAXCOVER* best attractive covers are returned at each iteration which accelerates convergence. If the genetic algorithm fails to find an attractive cover, it is applied afresh one more time before using the ILP. The stopping criterion that is used for our genetic algorithm is the maximum number of consecutive iterations it_{max} without improvement in best solution quality. However, the value of it_{max} varies over the set $OPTION = \{50, 100, 250, 500, 1000, 2000\}$ from one run to the other. If in the previous run genetic algorithm returned less than *MAXCOVER* attractive covers then it_{max} is set to next higher value, if possible, in the *OPTION*, otherwise it_{max} is set to next lower value, if possible, in the *OPTION*. it_{max} is set to 50 in the very first run of the genetic algorithm.

In case of LM-ASR, as many power levels are equal, therefore, whenever a sensor is set to a new power level a check is made to ensure that all targets that can be covered with that much power, are indeed covered.

4. Computational results

4.1. Instances and experimental setup

The instances that we have used in our computational experiments have been generated as follows. n sensors and m targets are generated at random in a 500×500 area. The maximum sensing range of sensors is set to $R^{max} = 150$. The instance name format is `nXXXmYYY`, where `XXX` $\in \{50, 100, 150, 200, 300, 600\}$ is the number of sensors, and `YYY` $\in \{15, 30, 45, 60, 90, 120, 180, 360\}$ is the number of targets. Five instances of the same size have been generated, leading to a grand total of 60 different instances.

In addition to LM-ASR, LM-PSR is addressed for three different numbers of sensing ranges. In PSR_1, sensors are either sensing up to their maximum range R^{max}, or not used at all; this corresponds to the problem where sensing ranges are not adjustable. In PSR_3, the three sensing ranges are $\{50, 100, 150\}$ and in PSR_6, the six considered sensing ranges are $\{25, 50, 75, 100, 125, 150\}$. Thus, the number of sensing ranges considered in this paper is as large as in [3]. The quadratic model f_{qua} defined in Section 2.1 has been used for the sensors' power consumption.

All the approaches presented in this paper are implemented in `C`, a function is used for removing the covers used a zero amount of time when the number of covers exceeds $10(n+m+1)$ in the master problem (this is expected to keep the master

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

program fast when *GA* returns up to *MAXCOVER* covers per iteration). Algorithm 2 is also run at every iteration for transforming the cover found by the auxiliary problem into a non-dominated cover.

For the genetic algorithm, we have used a population of $\min(n,100)$ individuals, $MAXCOVER=10$, $\pi_a=0.25$, $\pi_f=0.8$, $\pi_h=0.9$, $\pi_m=0.05$. We have used two different values of π_b. The first parent is selected with $\pi_b=0.9$, whereas the second parent is selected with $\pi_b=0.8$. All these parameter values are chosen empirically after large number of trials.

All the computations have been performed on an Intel Xeon Processor system at 2.66 GHz, with 8 GB RAM under Microsoft Windows 7. The LP and ILP solver is GLPK [10].

Table 3
Computation time with and without restricting the search for non-dominated covers.

n150m045	Average CPU time to optimality			
	PSR-1	PSR-3	PSR-6	ASR
Dominated covers	0.43	2.54	7.93	69.18
Non-dominated covers	0.44	2.17	6.89	52.82
Improvement (%)	−2.33	14.57	13.11	23.65

4.2. Results

As a preliminary result, the benefit of using non-dominated covers is assessed on the five instances with $n=150$ sensors and $m=45$ targets, for LM-ASR and LM-PSR. The first row of Table 3 displays the CPU time in seconds of the column generation algorithm for which the cover refinement procedure (see Algorithm 2) is never used, and where Eq. (18) is not present in the auxiliary problem. The second row of the table shows the results where the cover refinement procedure is called after each iteration of the auxiliary problem, and where Eq. (18) is present in the auxiliary problem formulation. In both cases, ILP only is used for addressing the auxiliary problem. The last row displays the CPU time improvement (in percent) when using non-dominated covers.

It can be seen that the benefit of restricting the search for attractive covers to non-dominated ones is more visible for LM-ASR, and for LM-PSR with a high number of predefined sensing ranges. These problems are the most difficult ones, as shown in Tables 4 and 5. Lifetime values are not reported in Table 3 because they are identical.

Table 4 reports the results where the auxiliary problem is addressed with ILP at each iteration, while Table 5 displays the results where *GA* is used for addressing the auxiliary problem, the ILP formulation being solved only when *GA* fails to find attractive covers.

Table 4
Lifetime and computation time when addressing the auxiliary problem with *ILP*.

Instances	Average network lifetime				Average CPU time to optimality			
	PSR-1	PSR-3	PSR-6	ASR	PSR-1	PSR-3	PSR-6	ASR
n050m015	4.000	7.621	9.561	12.262	$0.03^{(5)}$	$0.06^{(5)}$	$0.09^{(5)}$	$0.22^{(5)}$
n050m030	3.600	6.361	8.474	11.291	$0.03^{(5)}$	$0.08^{(5)}$	$0.16^{(5)}$	$0.37^{(5)}$
n100m030	9.000	19.487	24.089	31.665	$1.04^{(5)}$	$0.83^{(5)}$	$2.66^{(5)}$	$4.61^{(5)}$
n100m060	7.000	12.004	14.088	18.937	$0.96^{(5)}$	$0.96^{(5)}$	$2.23^{(5)}$	$178.66^{(5)}$
n150m045	11.800	24.360	32.502	44.934	$0.44^{(5)}$	$2.17^{(5)}$	$6.89^{(5)}$	$52.82^{(5)}$
n150m090	11.600	23.744	29.814	41.993	$1.18^{(5)}$	$11.26^{(5)}$	$126.39^{(5)}$	$95.29^{(2)}$
n200m060	14.600	32.378	46.867	64.455	$1.02^{(5)}$	$6.36^{(5)}$	$47.73^{(5)}$	$235.30^{(4)}$
n200m120	14.400	32.105	39.699	55.003	$2.81^{(5)}$	$20.70^{(5)}$	$67.83^{(5)}$	$270.79^{(2)}$
n300m090	25.000	55.525	74.139	103.320	$5.38^{(5)}$	$82.36^{(4)}$	$285.88^{(3)}$	$984.85^{(1)}$
n300m180	23.400	54.581	67.643	93.930	$14.14^{(5)}$	$1015.86^{(5)}$	$373.81^{(1)}$	$1\text{ h}^{(0)}$
n600m180	51.000	105.683	138.292	58.844	$82.34^{(5)}$	$722.41^{(3)}$	$963.55^{(1)}$	$1\text{ h}^{(0)}$
n600m360	48.600	96.000	90.625	16.176	$221.22^{(5)}$	$1\text{ h}^{(0)}$	$1\text{ h}^{(0)}$	$1\text{ h}^{(0)}$

Table 5
Lifetime and computation time when addressing the auxiliary problem with *ILP+GA*.

Instances	Average network lifetime				Average CPU time to optimality			
	PSR-1	PSR-3	PSR-6	ASR	PSR-1	PSR-3	PSR-6	ASR
n050m015	4.000	7.621	9.561	12.262	$0.01^{(5)}$	$0.02^{(5)}$	$0.03^{(5)}$	$0.08^{(5)}$
n050m030	3.600	6.361	8.474	11.291	$0.02^{(5)}$	$0.03^{(5)}$	$0.04^{(5)}$	$0.12^{(5)}$
n100m030	9.000	19.487	24.089	31.665	$0.03^{(5)}$	$0.47^{(5)}$	$1.41^{(5)}$	$3.53^{(5)}$
n100m060	7.000	12.004	14.088	18.937	$0.04^{(5)}$	$0.15^{(5)}$	$0.28^{(5)}$	$94.69^{(5)}$
n150m045	11.800	24.360	32.502	44.934	$0.05^{(5)}$	$0.29^{(5)}$	$3.48^{(5)}$	$25.43^{(5)}$
n150m090	11.600	23.744	29.814	41.994	$0.10^{(5)}$	$2.69^{(5)}$	$15.24^{(5)}$	$10.28^{(2)}$
n200m060	14.600	32.378	46.867	64.455	$0.08^{(5)}$	$0.29^{(5)}$	$21.03^{(5)}$	$520.87^{(5)}$
n200m120	14.400	32.105	39.699	55.015	$0.13^{(5)}$	$1.98^{(5)}$	$19.27^{(5)}$	$25.44^{(2)}$
n300m090	25.000	57.536	74.143	103.320	$0.20^{(5)}$	$26.73^{(4)}$	$101.59^{(3)}$	$934.02^{(2)}$
n300m180	23.400	54.581	67.643	94.239	$0.32^{(5)}$	$90.26^{(5)}$	$50.59^{(1)}$	$1\text{ h}^{(0)}$
n600m180	51.000	105.901	138.564	184.887	$1.01^{(5)}$	$114.24^{(3)}$	$44.34^{(1)}$	$1\text{ h}^{(0)}$
n600m360	48.600	96.156	122.299	153.522	$2.01^{(5)}$	$673.56^{(3)}$	$1\text{ h}^{(0)}$	$1\text{ h}^{(0)}$

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

Table 6
Summary comparison of *ILP* alone and *ILP*+*GA* for the auxiliary problem.

Problems	PSR-1	PSR-3	PSR-6	ASR
Number of optimal solutions found with *ILP*	60	52	45	34
Number of optimal solutions found with *ILP*+*GA*	60	55	45	36
CPU time cut with *ILP*+*GA* vs. *ILP* alone (for the instances solved to optimality with *ILP*)	98.78%	88.03%	79.69%	62.24%
Average lifetime with *ILP* on all instances	18.67	37.40	47.98	46.10
Average lifetime with *ILP*+*GA* on all instances	18.67	38.35	48.47	68.07

The first column displays the instance name. Columns 2–5 show the average lifetime for the five instances associated with each row for PSR_1, PSR_3, PSR_6 and ASR. Columns 6–9 display the average CPU time in seconds for returning an optimal solution. Each problem is allocated a maximum amount of 1 h per instance; the number of instances solved to optimality is shown in parenthesis. Consequently, the average CPU time is computed on these instances only.

Finally, Table 6 summarizes the results shown in Tables 4 and 5, by stressing the differences between addressing the auxiliary problem with the ILP formulation, and using the proposed hybrid approach. The table compares both approaches in terms of number of optimal solutions found, displays the CPU time cut provided by the hybrid approach compared to the ILP formulation, the average lifetime on all instances.

4.3. Discussion

Tables 4 and 5 show that for the same number of sensors, doubling the number of targets leads to a smaller lifetime. This is quite clear for LM-ASR that can take advantage of the actual distance between sensors and targets, whereas this is less visible for LM-PSR_1 for which the energy consumption of a sensor depends on R^{max} only. It can also be checked that maximum lifetimes are reached for ASR, which is an experimental verification of Lemma 3. The lifetime decreases when the number of sensing ranges decreases in LM-PSR, which was also a predictable result. The solution space of the auxiliary problem increases with the number of sensing ranges, hence LM-ASR is the most difficult problem and LM-PSR_1 is naturally the easiest one. Column generation algorithms are subject to the well-known tail effect: a very large number of iterations is required at the end of the search for proving optimality. Additionally, the last iterations (*i.e.*, the last generated columns) lead to a very modest increase in lifetime. Consequently, the lifetime found after 1 h is often very close to the optimal lifetime for $n \leq 300$. It is necessary to consider large instances for observing a significant gap in terms of solution quality: 1 h of computation time is not enough for solving LM-ASR on the set of instances n600m180 and n600m360, but it can be seen that using *GA* leads to much better results. Indeed, the instances must be sufficiently large for avoiding the tail effect of column generation to mask the gap between *GA* and *ILP*. For smaller instances, this gap is not visible because *ILP*, which is much slower than *GA*, is still in the tail of the search when the time limit of 1 h is elapsed. The benefit of using *GA* in terms of solution quality is more and more visible for difficult problems; this general trend is clearly visible in the last two rows of Table 6. At first sight, this table also suggests that the benefit of using *GA* is less visible for difficult problems like LM-ASR. But this is not the case: the benefit of using *GA* is clearly visible for large instances (see Tables 4 and 5), and these instances are not taken into account here as the CPU time cut is computed only for instances that are solved to optimality by both approaches. And as can be seen in Table 4, there are less and less large instances solved to optimality with *ILP* for difficult problems.

As stated in Lemma 3, the maximum lifetime is reached for LM-ASR. Naturally, when the number of predefined sensing ranges increases, the lifetime in LM-PSR converges toward its value in LM-ASR.

LM-PSR_1 is the easiest problem as sensors have only two states (sensing with range R^{max} or not sensing at all), LM-PSR is more and more difficult as the number of predefined sensing ranges increases. LM-ASR is the most difficult problem, which is not surprising as each sensor can have a large number of power levels if it covers a lot of targets. Furthermore, Eq. (20) ensures that the number of non-zero decision variables in the auxiliary problem in LM-PSR is less than in its counterpart in LM-ASR, whatever M_{PSR}.

5. Conclusions

In this paper we have proposed an exact approach for maximizing the lifetime of a wireless sensor network where sensing ranges of sensors can be adjusted. Two versions of the problem are considered. In the first version, sensing ranges can be continuously adjusted within the maximum sensing range, whereas in the second one, sensing ranges can be adjusted only within the set of predefined values. We have modelled these problems using a column generation scheme and devised a matheuristic (i.e., a combination of a metaheuristic within an exact approach) to efficiently solve the corresponding models. Theoretical results are provided so as to characterize efficient covers, and some relationships between the problems are also highlighted. The proposed approach addresses the auxiliary problem with an efficient genetic algorithm. Computational results demonstrate that for both versions, the use of genetic algorithm within the column generation scheme significantly reduces the computation time, without compromising optimality.

Our results suggest that the use of non-dominated covers could be efficiently adapted to different versions of the lifetime maximization problems. Similar approaches can be designed for other scheduling problems in wireless sensor networks. As a future work, we intend to experiment with other population based metaheuristics within the column generation scheme to assess their suitability for such use.

References

[1] Akyildiz I, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. Computer Networks 2002;38(4):393–422.
[2] Alfieri A, Bianco A, Brandimarte P, Chiasserini CF. Maximizing system lifetime in wireless sensor networks. European Journal of Operational Research 2007;181:390–402.
[3] Cardei M, Wu J, Lu M, Pervaiz M. Maximum network lifetime in wireless sensor networks with adjustable sensing ranges. In: IEEE international conference on wireless and mobile computing, networking and communications (WiMob'2005); 2005. p. 438–45.
[4] Chvátal V. Linear programming. New York: W. H. Freeman and Company; 1980.
[5] Davis L. Handbook of genetic algorithms. New York: Van Nostrand Reinhold; 1991.
[6] Dhawan A, Vu C, Zelikovsky A, Li Y, Prasad S. Maximum lifetime of sensor networks with adjustable sensing range. In: Seventh ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing; 2006. p. 285–9.
[7] FICO. Xpress-MP 〈http://www.dashoptimization.com/〉; 2009.

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

[8] Gentili M, Raiconi A. α-Coverage to extend network lifetime on wireless sensor networks. Optimization Letters (2012) 1–16.
[9] Gu Y, Ji Y, Li J, Zhao B. Theoretical treatment of target coverage problem in wireless sensor networks. Journal of Computer Science and Technology 2011;26:117–29.
[10] GLPK (GNU Linear Programming Kit) 〈http://www.gnu.org/software/glpk/〉; February 2012.
[11] Jia J, Chen J, Chang G, Wen Y, Song J. Multi-objective optimization for coverage control in wireless sensor network with adjustable sensing radius. Computers & Mathematics with Applications 2009;57:1767–75.
[12] Latré B, Braem B, Moerman I, Blondia C, Demeester P. A survey on wireless body area networks. Wireless Networks 2011;17:1–18.
[13] Lübbecke ME, Desrosiers J. Selected topics in column generation. Operations Research 2005;53:1007–23.
[14] Rossi A, Singh A, Sevaux M. A column generation algorithm for sensor coverage scheduling under bandwidth constraints. Networks, http://dx.doi.org/10.1002/net.20466, in press.
[15] Wang C, Thai MT, Li Y, Wang F, Wu W. Optimization scheme for sensor coverage scheduling with bandwidth constraints. Optimization Letters 2009;3(1):63–75.
[16] Wang J, Medidi S. Energy efficient coverage with variable sensing radii in wireless sensor networks. In: Third IEEE international conference on wireless and mobile computing, networking and communications (WiMob 2007). White Plains, New York, USA; 2007.
[17] Wu J, Yang S. Coverage issue in sensor networks with adjustable ranges. In: Proceedings of the international conference on parallel processing workshop, ICPP 2004; 2004. p. 61–8.
[18] Zhou Z, Das S, Gupta H. Variable radii connected sensor cover in sensor networks. ACM Transactions on Sensor Networks 2009;5(1):8:1–36.

Please cite this article as: Rossi A, et al. An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.04.001

Ann Oper Res (2011) 191:219–249
DOI 10.1007/s10479-011-0996-y

A sensitivity analysis to assess the completion time deviation for multi-purpose machines facing demand uncertainty

André Rossi · Alexis Aubry · Mireille Jacomino

Published online: 4 October 2011

Abstract This paper addresses multi-purpose machine configuration in an uncertain context through sensitivity analysis. The so-called configuration is the machine's ability to process products, and the uncertain context is modelled as a demand variation affecting the forecast demand. Given a configuration, this work aims at assessing the completion time deviation when the workshop demand is subject to perturbation. Such quantitative information can be used in a robustness approach for selecting the most appropriate configuration. To do so, the configuration impact on the completion time value that can be reached by solving the attached scheduling problem is first investigated. Then, the completion time deviation is written as a piecewise linear function of the magnitude of demand variation. The proposed approach, which is based on the solution of a set of linear programs, is illustrated through a detailed example. It is shown to be polynomial, and fast enough for addressing real-world instances. Finally, how to compare two configurations on the basis of completion time deviation in an uncertain context is demonstrated.

Keywords Sensitivity analysis · Multi-purpose machines · Linear programming · Configuration · Demand uncertainty

A. Rossi (✉)
Lab-STICC, Université de Bretagne-Sud, Centre de Recherche Christiaan Huygens, 56321 Lorient, France
e-mail: andre.rossi@univ-ubs.fr

A. Aubry
CRAN, Nancy-Université, Boulevard des Aiguillettes B.P. 70239, 54516 Vandœuvre lès Nancy, France
e-mail: alexis.aubry@cran.uhp-nancy.fr

M. Jacomino
G-SCOP, INPG, 46 avenue Félix Viallet, 38031 Grenoble, France
e-mail: mireille.jacomino@g-scop.inpg.fr

1 Introduction

Multi-purpose machines are used to model industrial workshops composed of parallel machines subject to constraints which prevent some machines from processing some product types (Brucker et al. 1997). These constraints may be technical and economical; they may be strict structural inability or be the result of a strategic choice. Setting the *configuration* of the workshop is choosing which machines are able to process which product types. the configuration has a significant impact on the completion time (also referred to as the makespan) of the products to be processed by the workshop (Leung and Li 2008).

In the semiconductor industry, the photolithography process is implemented by a set of parallel multi-purpose machines. It is known to be a challenging step as it involves machines that are very expensive and subject to rapid obsolescence due to the constant advance in that field (Akçali et al. 2001; Yoon and Lee 2004). Moreover, the production flow is not easy to manage as products may have complex reentrant routes, and the photolithography workshop is often observed to be a bottleneck. Finally, as the production processes are always updated to integrate new technologies, production volumes are hard to forecast because of unpredictable production losses.

This uncertain context is reflected by the occurrence of variations in the product mix and volume of the demand to be processed by the photolithography workshop. The decision-maker's concern is then to design the best possible configuration, i.e. to find a trade-off between the configuration cost and performance. The configuration cost is presented in the next section, its performance is measured by the completion time achieved by the best feasible schedule for this configuration.

The choice of the workshop's configuration in such a context is clearly a robustness issue: the decision-maker wants to build a low-cost configuration that ensures that high-quality schedules (i.e. having a low makespan) are feasible despite demand uncertainty. There are many definitions of robustness in the literature. Regarding scheduling (Billaut et al. 2008), robustness approaches are usually sorted into the two following classes:

- The first one is based on the optimization of a robustness criterion λ. One of the most popular definitions is due to Kouvelis and Yu (1997), it states that a solution $\mathcal{S}$ is robust if it optimizes the performance criterion z on a set $\mathcal{P}$ of instances that models uncertainty ($\mathcal{P}$ can be a discrete or a continuous set of instances). Finding a robust solution in the sense of Kouvelis et al. (1992) consists in minimizing one of the following robustness criteria:
  - Absolute robustness: $\lambda_1 = \max_{i\in\mathcal{P}} z_i(\mathcal{S})$, where $z_i(\mathcal{S})$ is the performance of solution $\mathcal{S}$ on instance i
  - Robust deviation: $\lambda_2 = \max_{i\in\mathcal{P}}(z_i(\mathcal{S}) - z_i^*)$, where z_i^* is the minimum criterion value for instance i over all the feasible solutions
  - Relative robustness: $\lambda_3 = \max_{i\in\mathcal{P}} \frac{z(\mathcal{S})-z_i^*}{z_i^*}$.

  In this paper, the solution $\mathcal{S}$ is the workshop's configuration, the performance criterion z is the makespan and $\mathcal{P}$ is the (continuous) set of demands over which the configuration is expected to be robust.
- The second approach states that a solution is robust if some conditions remain satisfied for all instances in $\mathcal{P}$. In Kouvelis et al. (1992), Snyder (2006), a solution is said to be p-robust on a set of instances $\mathcal{P}$ if the maximum gap between $z_i(\mathcal{S})$ and z_i^* is less than p. The $\mathcal{L}_\lambda$-robustness is also close to the definition of robustness introduced in Rossi (2003), Aubry et al. (2009), it states that a solution is robust if its robustness criterion value λ is less than a given threshold $\mathcal{L}_\lambda$. This definition appears to fit the framework in which

the decision-maker wants a robust solution to fit some requirements (modeled by the threshold $\mathcal{L}_\lambda$) on a risk to be covered (modeled by $\mathcal{P}$). In this definition a robust solution plays the role of an insurance to which the decision-maker subscribes. $\mathcal{P}$ represents the risk covered by the insurance and $\mathcal{L}_\lambda$ is the guarantee level of that insurance.

The generation of robust configurations is out of the scope of this paper. A sensitivity analysis (Penz et al. 2001) is performed to assess the completion time deviation as a piecewise linear function of the magnitude of demand uncertainty for a given configuration. In this respect, it can be used in the first class of robustness approaches if the decision-maker wants to find the configuration with the minimum makespan deviation over $\mathcal{P}$, or in the second class if one of the desired features for a robust configuration is to maintain a makespan deviation that is less than a given $\mathcal{L}_\lambda$ over $\mathcal{P}$. However, unlike most sensitivity analyses, this work does not focus on performance loss due to some schedule inappropriateness to the production context. Indeed, the underlying scheduling problem is supposed to be updatable at any time, and the completion time deviation due to the configuration inappropriateness to the actual demand only is investigated, where the so-called completion time is always the best possible one for the current configuration and the actual workshop demand. In such a context, performing a sensitivity analysis is meaningful and provides the decision-maker with a powerful tool for determining the trend of the behavior of a candidate configuration when the workshop has to cope with non-nominal demands. The considered uncertainties are additional quantities of products on the forecast demand. First, additional quantities of a single product type are considered. Then, the results are extended to additional quantities simultaneously affecting any product type.

This paper is organized as follows. The underlying scheduling problem is delineated in section two, and a procedure for computing the critical machines (i.e. the most loaded ones) along with some properties is shown in section three. When the solution to the scheduling problem has a particular structure (it is said to be r_k-split), the completion time value can be expressed as a function of a subset of the product types. Some properties and a procedure characterizing r_k-split production plans in such cases are provided in section four. These results are a prerequisite for performing a sensitivity analysis done in the fifth section, assuming that demand uncertainty affects a single product type. Finally, section six extends these results for addressing the problem of determining the completion time deviation as a non increasing piecewise linear function of the magnitude of uncertainty, and illustrates how the decision-maker can use the proposed sensitivity analysis for comparing two configurations. The results are illustrated throughout the paper with the use of a recurrent example, and discussed in the conclusion section.

2 The scheduling problem

The number of machines is denoted by m, and n is the number of product types. The vector N models the demand, N_i being the quantity of products of type i that is to be processed by the workshop. For some technological reasons, all the machines cannot always process all the product types. The processing actually becomes possible if some resources are added to the machine, and if some settings are performed. In that case, machine j is said to be *qualified* for the products of type i. The *configuration* of the workshop is a binary $n \times m$ matrix Q, such that $Q_{ij} = 1$ if and only if machine j is qualified for the product type i. A valid configuration must be such that at least one machine is qualified for each product type, and each machine is qualified for at least one product type. The cost of configuration Q is defined as the sum of all the qualification costs, it can be expressed as a time or as

a monetary amount. It should be noticed that the costliest configuration (if it is feasible) is defined by $Q_{ij} = 1$ for all $(i, j) \in \{1, \ldots, n\} \times \{1, \ldots, m\}$, which reduces to m parallel machines. In such a workshop, the decision-maker has to update the configuration matrix periodically in order to fit the forecast demand (which is also updated periodically), by ensuring that the underlying scheduling problem has a solution which meets a given deadline, even when the actual demand differs from the forecast demand. The typical updating period in a photolithography workshop is one month, it should not be too short because updating the configuration requires interrupting production and performing costly settings and tests on the machines. This is also the reason why the configuration cannot be adjusted on-the-fly.

The underlying scheduling problem involves the *speed matrix*, that is an $n \times m$ matrix W such that W_{ij} is the number of products of type i that machine j can process per unit of time (provided that $Q_{ij} = 1$). The machines are supposed to be uniform, i.e. W is a rank-one matrix, so it can be written as $W = P \times V$, where P is a strictly positive $n \times 1$ column vector, and V is a strictly positive $1 \times m$ row vector. This assumption is satisfied for most photolithography workshops.

Because the machines are very expensive and subject to rapid obsolescence, it makes sense to schedule the products to process so as to minimize the maximum completion time (this criterion is denoted by $C_{\max}$). The solution for such a problem is a value for $C_{\max}$ and R, where R is a production plan. More precisely, R_{ij} is equal to the time spent by machine j processing products of type i. Preemption and splitting are allowed in the problem that can be referred to as $QMPM|split|C_{\max}$ using the three-field notation (Graham et al. 1979), where *QMPM* stands for uniform multi-purpose machines, and *split* indicates that splitting and preemption are allowed.

This problem can be solved in polynomial time (Aubry et al. 2008a), and can be modelled as a linear program denoted by *LP*. This model is derived from Lawler and Labetoulle (1978) that has originally been designed to solve $R|pmtn|C_{\max}$. In this problem, machines are unrelated and splitting is not allowed, so $QMPM|split|C_{\max}$ can be viewed as a relaxation of $R|pmtn|C_{\max}$ restricted to uniform machines.

$$(LP): \begin{cases} \text{Minimize } C_{\max} & (1) \\ \sum_{j \in J} Q_{ij} P_i V_j R_{ij} = N_i \quad (\forall i \in I) & (2) \\ \sum_{i \in I} Q_{ij} R_{ij} \leq C_{\max} \quad (\forall j \in J) & (3) \\ R_{ij} \geq 0 \quad (\forall i \in I)\ (\forall j \in J) & (4) \end{cases}$$

Equation (1) is the objective function. The set of constraints (2) enforces that the demand is met for each product type. The set of constraints (3) enforces that the time workload of each machine is less than $C_{\max}$. Indeed, the load of any machine j for production plan R, denoted by $L_j(R)$, is equal to the sum of the $Q_{ij}R_{ij}$ over i in $I = \{1, \ldots, n\}$. Then, this set of constraints is sometimes written as $L_j(R) \leq C_{\max}$ for all j in $J = \{1, \ldots, m\}$. The set of constraints (4) ensures that the production plan is valid, as an amount of time cannot be negative. It should be stressed that the makespan $C_{\max}$ and the production plan R are the decision variables. The configuration Q, the speed matrix $W = P \times V$ and the demand N are given. In the rest of this paper, $C_{\max}(Q, N)$ denotes the optimal completion time value for configuration Q and demand N.

Once again, this paper aims at performing a sensitivity analysis for a given configuration, so for any demand, the production plan is supposed to be computed in a very short amount of CPU time. This is quite a realistic hypothesis as *LP* is a linear program, and to the best

of the authors' knowledge, industrial-size instances in the field of photolithography do not exceed $n = 50$ product types and $m = 50$ machines.

3 Finding critical machines

3.1 Balanced production plans

3.1.1 Definition

Solving *LP* leads to one of the two following situations. The production plan R is either *balanced*, or not: R is said to be balanced if all the machines share the same workload (i.e. $L_j(R) = C_{\max}(Q, N)$ for all j in J). Non balanced production plans are the result of some non-qualifications in the configuration matrix (i.e. zero entries in Q). It has been shown in Aubry et al. (2008a) that any balanced production plan is an optimal solution to *LP*, and that if there exists a balanced production plan for Q and N, then any optimal production plan to *LP* is also balanced. Consequently, if solving *LP* yields a non balanced production plan, then there does not exist any balanced optimal production plan for demand N.

3.1.2 Example

The following example is to be used intensively throughout this paper to illustrate the notions. Let us consider $m = 4$ machines for processing $n = 5$ product types. For the following configuration Q and speed matrix $W = P \times V$, the optimal production plan to *LP* for the demand N^a denoted by R^a is not balanced (this is due to the fact that machine 1 is qualified only for one product type) and leads to $C_{\max}(Q, N^a) = 12$. Demand N^b can also be observed to be such that R^b is balanced with $C_{\max}(Q, N^b) = 12$: the non-qualifications do not prevent load balancing in that case. Roughly speaking, it can be said that the configuration Q fits the demand N^b while it does not fit N^a. Moreover, if all qualifications are assumed to have identical unitary costs, then the cost of configuration Q is 10.

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \qquad P = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \qquad V = [1\ 1\ 1\ 1]$$

$$N^a = \begin{bmatrix} 5 \\ 5 \\ 10 \\ 10 \\ 16 \end{bmatrix}, \qquad R^a = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 2 & 0 & 8 \\ 0 & 0 & 12 & 4 \end{bmatrix}, \qquad N^b = \begin{bmatrix} 10 \\ 14 \\ 8 \\ 7 \\ 9 \end{bmatrix}$$

$$R^b = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 2 & 12 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 7 \\ 0 & 0 & 4 & 5 \end{bmatrix}$$

3.2 Critical machines and critical product types

3.2.1 Definitions and properties

Definition 1 Let Q be a configuration matrix and N be a demand.

- A machine is said to be critical if its load is equal to $C_{\max}(Q, N)$ for any optimal production plan to *LP*.
- A product type is said to be critical if all the machines that are qualified for it are critical.

Lemma 1 *Procedure*1(Q, R, I, J) *returns the set of critical machines* J^c *and the set of critical product types* I^c *from* Q *and any optimal production plan* R *for LP.*

Proof Let R be an optimal production plan to *LP* for Q and N. First, it is recalled that the numerical value of $C_{\max}(Q, N)$ can be deduced from R, and is common to any optimal production plan. By construction, *Procedure*1 returns a set of machines J^c and a set of product types I^c such that:

(a) $L_j(R) = C_{\max}(Q, N)$ for any j in J^c
Indeed, *Procedure*1 only removes machines from the set of those having a load equal to $C_{\max}$.

(b) $Q_{ij} = 0$ for any (i, j) in $I^c \times \bar{J}^c$ (with $\bar{J}^c = J \backslash J^c$)
This implies that $R_{ij} = 0$ for any (i, j) in $I^c \times \bar{J}^c$. *Procedure*1 only removes product types from the set of those that are only processable by critical machines.

Algorithm 1 A procedure for computing the critical machines and the critical product types

```
Procedure Procedure1(Q, R, I, J)
    [Initialization]
    J^c = {j ∈ J | L_j(R) = C_max(Q, N)}
    I^c = {i ∈ I | ∃j ∈ J^c, R_ij > 0}
    stable = 0
    While (stable == 0) do
        [Step 1: Finding I*, the set of the product types to be removed from I^c]
        [I* is initialized as the empty set]
        I* = ∅
        For i ∈ I^c do
            For j ∈ J\J^c do
                If (Q_ij == 1) then
                    I* = I* ∪ {i}
                end If
            end For
        end For
        [Step 2: Stability test and update of I^c and J^c]
        If (I* == ∅) then
            stable = 1
        else
            I^c = I^c\I*
            J^c = J^c\{j | i ∈ I*, R_ij > 0}
        end If
    done
End
```

(c) $R_{ij} = 0$ for any (i, j) in $\bar{I}^c \times J^c$ (with $\bar{I}^c = I \backslash I^c$)
Indeed, *Procedure*1 removes any machine j from J^c if it processes a product type that is also processed by a non critical machine.

This clearly shows that the numerical value for $C_{\max}(Q, N)$ only depends on the machines in J^c and on the product types in I^c. Since these machines are processing all the products of types in I^c (and only these products), the set of constraints (3) in *LP* can be written as:

$$\sum_{i \in I^c} Q_{ij} R_{ij} = C_{\max}(Q, N) \quad (\forall j \in J^c)$$

The set of constraints (2) in *LP* can be written as follows for any product type in I^c:

$$\sum_{j \in J^c} \left(Q_{ij} V_j R_{ij}\right) = \frac{N_i}{P_i} \quad (\forall i \in I^c)$$

Summing up the above equalities yields:

$$\sum_{j \in J^c} \left(V_j \sum_{i \in I^c} Q_{ij} R_{ij} \right) = \sum_{i \in I^c} \frac{N_i}{P_i}$$

As the sum of the $Q_{ij} R_{ij}$ over J^c is equal to $C_{\max}(Q, N)$, it can be deduced that:

$$C_{\max}(Q, N) = \frac{\sum_{i \in I^c} \frac{N_i}{P_i}}{\sum_{j \in J^c} V_j} \tag{5}$$

Now, it is proven by contradiction that J^c is the set of critical machines and that I^c is the set of critical product types.

It can be deduced from (b) that for any optimal production plan R, the set of constraints (2) in *LP* can be reformulated as follows:

$$\sum_{j \in J^c} \left(Q_{ij}.V_j.R_{ij}\right) = \frac{N_i}{P_i} \quad \left(\forall i \in I^c\right)$$

Summing up all the above equalities yields the following equation referred to as (A):

$$\text{(A)} \quad \sum_{j \in J^c} V_j . \sum_{i \in I^c} \left(Q_{ij}.R_{ij}\right) = \sum_{i \in I^c} \frac{N_i}{P_i}$$

By definition of L_j, it can be deduced that any optimal production plan R is such that:

$$L_j(R) = \sum_{i \in I} \left(Q_{ij} R_{ij}\right) \geq \sum_{i \in I^c} \left(Q_{ij} R_{ij}\right) \quad \left(\forall j \in J\right)$$

The following inequality denoted by (B) can be deduced from equation (A) and the above inequality:

$$\text{(B)} \quad \sum_{j \in J^c} V_j . L_j(R) \geq \sum_{i \in I^c} \frac{N_i}{P_i}$$

It is stressed that the inequality (B) holds for any optimal production plan R.

Finally, it is assumed that there exists an optimal production plan R' such that machine γ in J^c has a load that is less than $C_{\max}(Q,N)$. More formally, R' is such that:

$$\begin{cases} L_j(R') \leq C_{\max}(Q,N) & (\forall j \in J \backslash \{\gamma\}) \\ L_\gamma(R') < C_{\max}(Q,N) \end{cases}$$

Computing the sum of the $V_j.L_j(R')$ for all j in J^c leads to:

$$\sum_{j\in J^c} V_j.L_j(R') < \sum_{j\in J^c} V_j.C_{\max}(Q,N)$$

Replacing $C_{\max}(Q,N)$ with its expression yields:

$$\sum_{j\in J^c} V_j.L_j(R') < \sum_{i\in I^c} \frac{N_i}{P_i}$$

This inequality is in contradiction with the inequality (B). This shows that for all j in J^c, there does not exist any optimal production plan R such that $L_j(R) < C_{\max}(Q,N)$. So J^c is the set of critical machines and, because of (b), I^c is the set of critical product types. □

Lemma 2 *J^c and I^c are nonempty.*

Proof Let us assume that J^c is empty. This means that for any machine γ in J there exists an optimal production plan denoted by R^γ such that $L\gamma < C_{\max}(Q,N)$. Let R^{sum} be the production plan defined by:

$$R_{ij}^{sum} = \frac{1}{m}.\sum_{\gamma\in J} R_{ij}^{\gamma} \quad \left(\forall i \in I\right)\left(\forall j \in J\right)$$

It can easily be seen that R^{sum} satisfies all the constraints of *LP*. Moreover, it is such that $L_j(R^{sum}) < C_{\max}(Q,N)$ for all j in J. This indicates that the objective function value for R^{sum} is strictly less than $C_{\max}(Q,N)$. This is a contradiction since $C_{\max}(Q,N)$ is the minimum value for the objective function of *LP*.

Now, it is shown by contradiction that I^c is also nonempty. Let us assume that I^c is empty, which means that all the product types that a critical machine can process are also processable by a non critical machine. In particular, for any optimal production plan R, any product type i that is actually processed by a critical machine j is also processable by a non critical machine j'. More formally, for all product types i such that $R_{ij} > 0$, there exists a non critical machine j' such that $Q_{ij'} = 1$ and $L_{j'}(R) < C_{\max}(Q,N)$. This clearly shows that it is possible to build a new optimal production plan from R, by transferring some sufficiently small quantity of products of type i from j to j', so that $L_j(R)$ and $L_{j'}(R)$ are strictly less than $C_{\max}(Q,N)$. This is a contradiction. □

It can be noticed that if there exists a balanced optimal production plan for Q and N, then $J^c = J$, $I^c = I$ and any optimal production plan to *LP* is balanced. In such a case, according to (5), the completion time optimal value is denoted by $C^b(N)$ and is defined by:

$$C^b(N) = \frac{\sum_{i\in I} \frac{N_i}{P_i}}{\sum_{j\in J} V_j}$$

More generally, whether an optimal production plan is balanced or not, (5) clearly shows that the makespan only depends on I^c and J^c. As a consequence, non critical machines and non critical product types can be disregarded when computing the completion time for non balanced production plans.

3.2.2 *Computational complexity analysis of Procedure*1

In all the sections devoted to computational analysis (namely the present section, Sects. 4.3.2 and 5.5), it is assumed that $n \geq m$ which is mostly the case in practice. The reader should replace n by m in the complexity results when $n < m$. For the sake of readability, it has been chosen not to use an expression like $\max(n, m)$ even though it would avoid making any assumption on n and m.

Regarding the complexity of *Procedure*1, it can be noticed that the While loop is executed at most n times as I^c, whose cardinality is in $\{1, \ldots, n\}$, is removed at least one element per iteration of that loop (except for the last one in which *stable* is set to 1). Inside that loop, *Step 1* performs $|I^c| \times (m - |J^c|)$ comparisons, i.e. less than $n \times m$ operations, and *Step 2* performs at most 3 operations. Consequently, the overall complexity of *Procedure*1 is $\mathcal{O}(n^3)$ operations.

3.2.3 *Example*

Let N^c be the demand defined below with the configuration matrix Q and the speed vectors P and V introduced in Sect. 3.1.2. The two following production plans R^c and R'^c are optimal for N^c, with $C_{\max}(Q, N^c) = 10$.

$$N^c = \begin{bmatrix} 3 \\ 3 \\ 2 \\ 6 \\ 20 \end{bmatrix}, \qquad R^c = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 10 & 10 \end{bmatrix}, \qquad R'^c = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 10 & 10 \end{bmatrix}$$

In R^c, the load of machines 2, 3 and 4 is equal to $C_{\max}(Q, N^c)$. However, invoking *Procedure*1(Q, R^c, I, J) returns $J^c = \{3, 4\}$ and $I^c = \{5\}$. Indeed, it can easily be seen that machine 2 can transfer some quantity of product of type 2 to machine 1 while keeping the load of these machines strictly less than 10.

In R'^c, machines 3 and 4 only have a load equal to $C_{\max}(Q, N^c)$. It is not surprising as machine 2 is not critical. Invoking *Procedure*1(Q, R'^c, I, J) yields the same result for J^c and I^c, but runs faster as the stability test is immediately passed. Indeed, the critical machines clearly appear in R'^c.

Finally, knowing J^c and I^c, it can be checked that:

$$C_{\max}(Q, N^c) = \frac{\sum_{i \in I^c} \frac{N_i^c}{P_i}}{\sum_{j \in J^c} V_j} = \frac{\frac{20}{1}}{1+1} = 10$$

It can be observed that as R^c and R'^c are not balanced, the makespan value is independent of product types $I \backslash I^c = \{1, 2, 3, 4\}$ and of machines $J \backslash J^c = \{1, 2\}$.

4 Expressing the completion time for r_k-split production plans

It has been shown in Sect. 3 that critical machines and critical product types alone are sufficient for expressing the completion time of an optimal production plan. Consequently, when studying the makespan, any non balanced production plan can be reduced to a balanced one by ignoring non critical machines and non critical product types. That is the reason why it is assumed in this section that all the production plans are balanced without loss of generality, as they are restricted to the product types in I^c and to the machines in J^c.

The focus is now put on I^c and J^c for showing that the makespan can sometimes be written using a strict subset of critical product types and a strict subset of critical machines. Such a situation occurs when the production plan is said to be r_k-*split*: in which case I^c and J^c are partitioned on the basis of one or more so-called *maximum rectangles of zeros* denoted by r_k in the configuration matrix. These notions are defined in Sect. 4.1. Then, it is shown how the makespan can be written as a function of a subset of product types and as a subset of machines, based on the maximum rectangles of zeros of the configuration matrix in Sect. 4.2. The main objective of this section is then addressed: for a given critical product type y, we want to write the makespan as a function of a minimum-cardinality subset of critical machines and of a minimum-cardinality subset of critical product types that contains y. These subsets are said to be *unsplittable* and are denoted by I^y and J^y. They are defined in Sect. 4.3 and serve as basic elements upon which the sensitivity analysis is grounded, as they allow one to assess the makespan deviation when the amount of products of type y is considered for increase.

4.1 r_k-split production plans

4.1.1 Definitions

Definition 2 The maximum rectangle of zeros in Q denoted by r_k is the Cartesian product $I^k \times J^k$, such that:

- $Q_{ij} = 0$ for all i in I^k and for all j in J^k
- There does not exist i' in $\bar{I}^k = I \backslash I^k$ such that $Q_{i'j} = 0$ for all j in J^k
- There does not exist j' in $\bar{J}^k = J \backslash J^k$ such that $Q_{ij'} = 0$ for all i in I^k.

Finding all the maximum rectangles of zeros in a configuration Q can be achieved by running an algorithm given in Espinouse et al. (2008), which is an adaptation of an original algorithm by Nourine and Raynaud (1999) initially designed for data mining. This algorithm may be unpracticable for large instances as there can be up to $2^n - 2$ such maximum rectangles of zeros if Q is equal to the identity matrix of dimension n. Fortunately, while based on the maximum rectangles of zeros in Q, the present work does not require computing them all.

Definition 3 A production plan R is said to be r_k-split if there exists a maximum rectangle of zeros in Q denoted by $r_k = I^k \times J^k$ such that $R_{ij} = 0$ for all (i, j) in $\bar{I}^k \times \bar{J}^k$.

It is recalled that R is defined on the product types in I^c and on the machines in J^c, so it is balanced, $I^k \subset I^c$ and $J^k \subset J^c$. Moreover, it satisfies $R_{ij} = 0$ for all (i, j) in $I^k \times J^k$ since $Q_{ij} = 0$ for all (i, j) in $I^k \times J^k$.

4.1.2 Example

In the example introduced in Sect. 3.1.2, it can be seen that $\{1\} \times \{2,3,4\}$ is a maximum rectangle of zeros in Q. However, it should be stressed that the "rectangles" are not necessarily connected: for instance, $\{1,5\} \times \{2\}$ is also a maximum rectangle of zeros. These rectangles can be arbitrarily numbered, even if all of them do not have to be enumerated in practice. However, this enumeration is performed on Q for the sake of the example, in which there exist seven maximum rectangles of zeros:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \quad \begin{cases} r_1 = \{1\} \times \{2,3,4\}, \\ r_2 = \{1,2,3\} \times \{4\}, \\ r_3 = \{1,5\} \times \{2\}, \\ r_4 = \{3\} \times \{1,4\}, \\ r_5 = \{3,4,5\} \times \{1\}, \\ r_6 = \{5\} \times \{1,2\}, \\ r_7 = \{1,2\} \times \{3,4\}, \end{cases} \quad R^b = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 2 & 12 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 7 \\ 0 & 0 & 4 & 5 \end{bmatrix}$$

With this numbering, it can be observed that the production plan R^b is r_7-split (rectangle r_7 appears in gray). Indeed, $R^b_{ij} = 0$ for all (i, j) in $\{3,4,5\} \times \{1,2\}$ and R^b also satisfies $R^b_{ij} = 0$ for all (i, j) in $r_7 = \{1,2\} \times \{3,4\}$.

In the next section, two lemmas are proven to show that the completion time value $C_{\max}(Q, N)$ can be written as a function of a strict subset of I^c and of a strict subset of J^c if there exists a maximum rectangle of zeros in Q such that an optimal production plan is r_k-split.

4.2 Expressing the completion time

4.2.1 Characterization and properties for r_k-split production plans

Lemma 3 *Let $r_k = I^k \times J^k$ be a maximum rectangle of zeros in matrix Q. There exists a r_k-split production plan for demand N if and only if the following equality holds.*

$$\text{(C)} \quad \frac{\sum_{i \in I^k} \frac{N_i}{P_i}}{\sum_{j \in \bar{J}^k} V_j} = \frac{\sum_{i \in \bar{I}^k} \frac{N_i}{P_i}}{\sum_{j \in J^k} V_j} = C^b(N)$$

It is recalled that $C^b(N)$ is the makespan value for any balanced production plan.

Proof First, it is shown that if R is r_k-split, then equation (C) holds. It yields from the set of constraints (3) in (LP) that $L_j(R) = \sum_{i \in I} Q_{ij} R_{ij} = C^b(N)$ for all j in J since R is a balanced production plan. Then, it can be written that:

$$\begin{cases} \sum_{j \in \bar{J}^k} (L_j(R) V_j) = \sum_{i \in I} \sum_{j \in \bar{J}^k} (Q_{ij} R_{ij} V_j) \\ \sum_{j \in J^k} (L_j(R) V_j) = \sum_{i \in I} \sum_{j \in J^k} (Q_{ij} R_{ij} V_j) \end{cases}$$

By definition of a r_k-split production plan, the machines in $\bar{J}^k$ exclusively process the products of types I^k and the machines in J^k exclusively process the products of types $\bar{I}^k$.

The expression of their load $L_j(R)$ can then be re-expressed accordingly.

$$\begin{cases} \sum_{j\in\bar{J}^k}(L_j(R)V_j) = \sum_{i\in I^k}\sum_{j\in\bar{J}^k}(Q_{ij}R_{ij}V_j) \\ \sum_{j\in J^k}(L_j(R)V_j) = \sum_{i\in\bar{I}^k}\sum_{j\in J^k}(Q_{ij}R_{ij}V_j) \end{cases}$$

The set of constraints (2) in (LP) can also be reformulated as follows:

$$\begin{cases} \sum_{j\in\bar{J}^k}(Q_{ij}R_{ij}V_j) = \frac{N_i}{P_i} \quad (\forall i \in I^k) \\ \sum_{j\in J^k}(Q_{ij}R_{ij}V_j) = \frac{N_i}{P_i} \quad (\forall i \in \bar{I}^k) \end{cases}$$

And as $L_j(R) = C^b(N)$, this immediately leads to equation (C).

Second, the reciprocal implication is shown by contraposition, i.e. it is shown that if there exists (i', j') in $\bar{I}^k \times \bar{J}^k$ such that $R_{i'j'} > 0$, then equation (C) does not hold.

Since i' is in $\bar{I}^k$, it can be deduced that $L_{j'}(R)$ is strictly greater than the load of the machine j' computed on I^k only. It can also be deduced that the demand for the products of type i' is not completely covered by the machines in J^k:

$$\begin{cases} \sum_{i\in I^k}(Q_{ij'}R_{ij'}) < \sum_{i\in I}(Q_{ij'}R_{ij'}) = L_{j'}(R) \\ \sum_{j\in J^k}(Q_{i'j}R_{i'j}V_j) < \sum_{j\in J}(Q_{i'j}R_{i'j}V_j) = \frac{N_{i'}}{P_{i'}} \end{cases}$$

Computing the sum of the $L_j(R)V_j$ for all j in $\bar{J}^k$ from the first inequality and the sum of the $\frac{N_i}{P_i}$ for all i in $\bar{I}^k$ from the second one yields:

$$\begin{cases} \sum_{i\in I^k}\sum_{j\in\bar{J}^k}(Q_{ij}R_{ij}V_j) < \sum_{j\in\bar{J}^k}(L_j(R)V_j) \\ \sum_{j\in J^k}\sum_{i\in\bar{I}^k}(Q_{ij}R_{ij}V_j) < \sum_{i\in\bar{I}^k}\frac{N_i}{P_i} \end{cases}$$

In the first inequality, $L_j(R)$ is replaced by $C^b(N)$ and $\sum_{j\in\bar{J}^k}(Q_{ij}R_{ij}V_j)$ is replaced by $\frac{N_i}{P_i}$. In the second one, $L_j(R) = \sum_{i\in\bar{I}^k}(Q_{ij}R_{ij})$ for all j in J^k is replaced by $C^b(N)$:

$$\begin{cases} \sum_{i\in I^k}\frac{N_i}{P_i} < C^b(N)\sum_{j\in\bar{J}^k}V_j \\ C^b(N)\sum_{j\in J^k}V_j < \sum_{i\in\bar{I}^k}\frac{N_i}{P_i} \end{cases}$$

This leads to:

$$\frac{\sum_{i\in I^k}\frac{N_i}{P_i}}{\sum_{j\in\bar{J}^k}V_j} < C^b(N) < \frac{\sum_{i\in\bar{I}^k}\frac{N_i}{P_i}}{\sum_{j\in J^k}V_j}$$

Thus, equation (C) does not hold. □

The following lemma is used for proving that if a single optimal production plan is r_k-split, then any optimal production plan also has to be r_k-split. This property is important because the result of the procedure for computing unsplittable product types and unsplittable machines in Sect. 4.3 only depends on the r_k-split feature of the production plan which has to be passed to that procedure.

Lemma 4 *Let $r_k = I^k \times J^k$ be a maximum rectangle of zeros in matrix Q. If there exists an optimal r_k-split production plan for N, then any optimal production plan for N is r_k-split.*

Proof This lemma is shown by contradiction. It is assumed that there exists a r_k-split optimal production plan R, and an optimal production plan R' that is not r_k-split (i.e. there exists $(i', j') \in \bar{I}^k \times \bar{J}^k$ such that $R'_{i'j'} > 0$). Following Lemma 3, the completion time for R can be written as:

$$C^b(N) = \frac{\sum_{i \in I^k} \frac{N_i}{P_i}}{\sum_{j \in \bar{J}^k} V_j}$$

Because $R'_{i'j'} > 0$, a strict inequality can be deduced on the load of machine j' for production plan R':

$$L_{j'}(R') = \sum_{i \in I} Q_{ij'} R'_{ij'} > \sum_{i \in I^k} Q_{ij'} R'_{ij'}$$

Computing the sum of the $L_j(R')V_j$ for all j in $\bar{J}^k$ yields:

$$\sum_{j \in \bar{J}^k} L_j(R')V_j > \sum_{j \in \bar{J}^k} \sum_{i \in I^k} Q_{ij} R'_{ij} V_j = \sum_{i \in I^k} \sum_{j \in \bar{J}^k} Q_{ij} R'_{ij} V_j$$

Since $I^k \times J^k$ is a maximum rectangle of zeros in Q, the set of constraints (2) in *LP* can be written as:

$$\sum_{j \in \bar{J}^k} Q_{ij} R'_{ij} V_j = \frac{N_i}{P_i} \quad (\forall i \in I^k)$$

Reporting this result in the last inequality leads to:

$$\sum_{j \in \bar{J}^k} L_j(R')V_j > \sum_{i \in I^k} \frac{N_i}{P_i}$$

Finally, as R' is a balanced production plan, $L_j(R')$ can be replaced by $C^b(N)$ for all j in $\bar{J}^k$, leading to:

$$C^b(N) > \frac{\sum_{i \in I^k} \frac{N_i}{P_i}}{\sum_{j \in \bar{J}^k} V_j}$$

Contradiction. □

4.2.2 Example

The example introduced in Sect. 3.1.2 is used once more to illustrate Lemmas 3 and 4. The production plan R^b has been shown to be r_7-split, with $r_7 = \{1, 2\} \times \{3, 4\}$. A different production plan for demand N^b denoted by R'^b can be found:

$$R'^b = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 2 & 12 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 9 \end{bmatrix}$$

It can easily be seen that R'^b is also r_7-split (with $I^7 = \{1, 2\}$ and $J^7 = \{3, 4\}$).

Furthermore, the numerical value for $C^b(N^b)$ can be written as:

$$C^b(N^b) = \frac{\frac{10}{1} + \frac{14}{1}}{1+1} = \frac{\frac{8}{1} + \frac{7}{1} + \frac{9}{1}}{1+1} = 12$$

4.3 Unsplittable sets of product types and machines

As shown in Lemma 3, the completion time can be written as a strict subset of I^c and J^c if the optimal production plan is r_k-split. In this section, we search for the unsplittable set of product types I^y and the unsplittable set of machines J^y for a given product type $y \in I^y$ in such a way that the production plan reduced to $I^y \times J^y$ is not r_k-split.

For any critical product y, *Procedure*2 shown in Algorithm 2 returns the unsplittable sets I^y and J^y as proven in Lemma 5. These sets are computed because when the amount of products of type y increases slightly, I^y and J^y become critical (as shown in Lemma 6), and since critical product types and critical machines allow for calculating the completion time, I^y and J^y play a key role for assessing the makespan deviation when the demand for the products of type y increases. Lemmas 7 and 8 are a prerequisite to this end.

4.3.1 Definition and properties

Definition 4 Let y be a product type in I^c. The set of product types I^y and the set of machines J^y are said to be *unsplittable* for y if:

1. y belongs to I^y
2. No r_k-split optimal production plan can be found with $y \in I^k$ when solving *LP* restricted to the product types in I^y and the machines in J^y.

It must be stressed that when considering *LP* restricted to $I^y \times J^y$, then the definition for r_k-split relies on the maximum rectangles of zeros in Q and R also restricted to I^y and J^y. Furthermore, the returned sets I^y and J^y do not depend on the production plan R passed to *Procedure*2. Indeed, the only hypothesis on R is that it is r_k-split, and all optimal production plans share that feature as shown in Lemma 4.

Lemma 5 *Let I^y and J^y be yielded by invoking Procedure*2(y, Q, R, I, J). *I^y and J^y are unsplittable.*

Proof The lemma is proven by contradiction. It is assumed that I^y and J^y are such that R is r_k-split on $I^y \times J^y$, with $y \in I^k$. It can be observed in *Procedure*2 that during the first execution of *Step 1*, J^y is set to S^y, the set of machines that are qualified for the product type y. As a consequence, $I^y \subset I^k$ and $J^y \subset \bar{J}^k$, which also means that $I^y \cap \bar{I}^k$ and $J^y \cap J^k$ are empty sets.

Since the final J^y (i.e. when *Procedure*2 terminates) must contain J^k, it can be deduced that at some iteration of *Procedure*2, one machine $j \in J^k$ is added to J^y. The condition for j to be added to J^y is that at least one product type $i \in I^y$ is processable by machine j. However, since R is r_k-split, $Q_{ij} = 0$ for all $(i, j) \in I^k \times J^k$. So, no machine in J^k can be added to J^y as long as $I^y \cap \bar{I}^k$ is empty.

This leads to examine the conditions under which a product type $i \in \bar{I}^k$ is added to I^y in *Procedure*2. It can be seen in *Step 2* that such an addition happens only if at least one machine in J^y processes a nonzero amount of product of type i. However, since R is r_k-split, $R_{ij} = 0$ for all $(i, j) \in \bar{I}^k \times \bar{J}^k$. So, no product type in $\bar{I}^k$ can be added to I^y as long as $J^y \cap J^k$ is empty.

Algorithm 2 A procedure for computing the unsplittable set of products and the unsplittable set of machines

```
Procedure Procedure2(y, Q, R, I, J)
    [Initialization]
    I^y = {y}
    J^y = ∅
    stable = 0
    While (stable == 0) do
        stable = 1
        [Step 1: Updating J^y]
        For i ∈ I^y do
            For j ∈ J\J^y do
                If (Q_ij == 1) then
                    J^y = J^y ∪ {j}
                    stable = 0
                end If
            end For
        end For
        [Step 2: Updating I^y]
        For j ∈ J^y do
            For i ∈ I\I^y do
                If (R_ij > 0) then
                    I^y = I^y ∪ {i}
                    stable = 0
                end If
            end For
        end For

    done
End
```

This shows that the condition for expanding J^y to J^k is that I^y is expanded to $\bar{I^k}$, and that the condition for expanding I^y to I^k is that J^y is expanded to J^k. Since none of these conditions is satisfied in *Procedure*2 at the point where $I^y = \{y\}$ and $J^y = S^y$, I^y and J^y are proven to be unsplittable. □

Lemma 6 *Let I^y and J^y be yielded by invoking Procedure*2(y, Q, R, I, J). *If the demand for the products of type y denoted by N_y increases by an infinitesimal amount $\Delta N^y > 0$, then J^y are the critical machines and I^y are the critical product types.*

Proof Let j be a machine in J^y. This machine has been added to J^y at *Step 1* in *Procedure*2 because it is qualified for product type i. If $i = y$, then machine j is in S^y and its load increases to compensate for the increase of N_y by ΔN^y. If $i \neq y$, then product type i has been added to I^y at *Step 2* in *Procedure*2 because there exists a machine j' in J^y that is processing a nonzero quantity of product of type i (i.e. $R_{ij'} > 0$). Thus, it is possible to move a nonzero quantity of products of type i from machine j' to machine j. The same reasoning can be applied to machine j' (j' belongs to J^y because it is qualified for product type $i' \neq i$ etc.) and thus, it can be deduced that an additional quantity of products of type y leads to an increase of the load of all the machines in J^y provided that ΔN^y is sufficiently small. Furthermore, the machines that do not belong to J^y cannot help with an additional quantity of product of type y because no load can be transfered to them (otherwise, they

would belong to J^y), consequently their load is strictly less than the load of the machines in J^y, so the machines in $J \backslash J^y$ are not critical. □

Lemma 7 *Let I^y and J^y be yielded by invoking Procedure*2(y, Q, R, I, J). *The optimal completion time value can be written as*:

$$C_{\max}(Q, N) = \frac{\sum_{i \in I^c} \frac{N_i}{P_i}}{\sum_{j \in J^c} V_j} = \frac{\sum_{i \in I^y} \frac{N_i}{P_i}}{\sum_{j \in J^y} V_j}$$

Proof By hypothesis, R is balanced so equality (5) holds. Let j be a machine in J^y. By construction of *Procedure*2, all the products processed by machine j belong to I^y (see *Step 2*). Moreover, all the machines qualified for these product types also belong to J^y (see *Step 1*). Let i be a product type in I^y. All the machines qualified for this product type belong to J^y. Consequently, R restricted to I^y and J^y is a balanced production plan, so I^y and J^y are also critical. This proves the second equality. □

Lemma 8 *If $I^y \neq I^c$ or $J^y \neq J^c$, then R is r_k-split.*

Proof First, the following equivalence is proven: $I^y \neq I^c$ if and only if $J^y \neq J^c$. If $I^y \neq I^c$, there exists at least one product type that is not processed by the machines in J^y. As a consequence, there exists at least one machine in $J^c \backslash J^y$ processing this product type. Conversely if $J^y \neq J^c$ there exists at least one machine that is processing at least one product type that is not in I^y (otherwise this machine would have a zero load which is impossible since R is supposed to be balanced).

If $J^y \neq J^c$, then no machine in $J^c \backslash J^y$ is qualified for any product type in I^y, i.e. $Q_{ij'} = 0$ for all $(i, j') \in I^y \times (J^c \backslash J^y)$. By construction of *Procedure*2, any product type i' that does not belong to I^y is such that $R_{i'j} = 0$ for all $(i', j) \in (I^c \backslash I^y) \times J^y$. This shows that R is r_k-split. □

*4.3.2 Computational complexity analysis of Procedure*2

Regarding the complexity of *Procedure*2, it can be noticed that the While loop is executed at most $n + m$ times. Indeed, initially $|I^y| = 1$ and $|J^y| = 0$ and these sets can grow to cardinality n and m respectively. Moreover, except for its last iteration, the While loop is such that one element is added to either I^y or J^y (or both). Inside that loop, *Step 1* performs $|I^y| \times (m - |J^y|)$ conditional instructions and *Step 2* performs $|J^y| \times (n - |I^y|)$ conditional instructions. Thus, it can be shown that the total number of conditional instructions is always less than $\frac{nm}{2}$ per iteration of the While loop. Consequently, the overall complexity of *Procedure*2 is $\mathcal{O}(n^3)$ operations.

4.3.3 Example

A larger example is proposed to illustrate unsplittable sets of product types and of machines. Let $n = 6$ product types and $m = 5$ machines having one-speed matrix W. The configuration

Q', the demand N' and an optimal production plan R' are shown below.

$$Q' = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \qquad N' = \begin{bmatrix} 7 \\ 13 \\ 10 \\ 8 \\ 2 \\ 10 \end{bmatrix}, \qquad R' = \begin{bmatrix} 0 & 7 & 0 & 0 & 0 \\ 10 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 7 & 0 \\ 0 & 0 & 5 & 3 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 \end{bmatrix}$$

It can be observed that R' is r_1-split with $r_1 = \{6\} \times \{1, 2, 3, 4\}$ (r_1 is surrounded with a dashed line). According to Lemma 3, the completion time can be written as

$$C^b(N') = \frac{\frac{N'_6}{P_6}}{V_5} = \frac{\sum_{i\in\{1,2,3,4,5\}} \frac{N'_i}{P_i}}{\sum_{j\in\{1,2,3,4\}} V_j} = 10$$

This shows that the completion time value can be written as a function of N'_4, but the formula also involves N'_1, N'_2, N'_3 and N'_5. Let us arbitrarily focus on the product type $y = 4$. Invoking *Procedure*2$(4, Q, R', I, J)$ with $I = \{1, \ldots, 6\}$ and $J = \{1, \ldots, 5\}$ yields the unsplittable set of products $I^4 = \{3, 4, 5\}$ and of machines $J^4 = \{3, 4\}$. Indeed, R' is also r_2-split, with $r_2 = \{3, 4, 5\} \times \{1, 2, 5\}$ (r_2 is shown in gray in R'). Thus, $C^b(N')$ can also be expressed as follows:

$$C^b(N') = \frac{\sum_{i\in\{3,4,5\}} \frac{N'_i}{P_i}}{\sum_{j\in\{3,4\}} V_j} = 10$$

Since R' restricted to the product types in I^4 and to the machines in J^4 is not r_k-split, the above formula for $C^b(N')$ is the one involving N'_4 and the fewest other product types. Furthermore, it can be said, using Lemma 6, that if N^4 increases then the machines j with $j \in \{3, 4\}$ are the only ones whose load increases, since it is not possible to transfer their load to other machines in the workshop. Thus, the results on r_k-split production plans are used in the next section to assess the completion time deviation when the demand varies.

5 Makespan deviation for a single product type

5.1 Modelling demand uncertainty

This section focuses on the makespan deviation when an additional quantity of products of type y affects the forecast demand. Unlike in Sect. 4, production plans are no longer assumed to be balanced, so any product type, whether critical or not, is separately considered for increase. The simultaneous increase of some product types is addressed in Sect. 6.

More formally, the purpose of this section is to assess the makespan deviation $\Delta C^y_{\max}$ for any nonnegative additional quantity of products of type y denoted by ΔN^y, from the forecast demand N^*. Thus,

$$\Delta C^y_{\max}(\Delta N^y) = C_{\max}(Q, N^y) - C_{\max}(Q, N^*)$$

where N^y is the demand resulting from an increase of ΔN^y products of type y, defined as follows:

$$\begin{cases} N^y_i = N^*_i \quad (\forall i \in I \backslash \{y\}) \\ N^y_y = N^*_y + \Delta N^y \end{cases}$$

The approach for determining the makespan deviation relies on $p+1$ successive steps, where p is an integer that will be shown to be less than m.

5.2 Approach overview

5.2.1 General framework

The proposed approach for determining the makespan deviation for all $\Delta N^y \geq 0$ can be presented as a flowchart (see Fig. 1). It relies on two stages.

First, the maximum quantity of products of type y that can be added to the forecast demand N^* while keeping the makespan value unchanged is denoted by a_0 and computed by solving a linear program called LP_0. Then, it can be deduced that $\Delta C^y_{\max}(\Delta N^y) = 0$, for all $\Delta N^y \leq a_0$ (where a_0 may possibly be zero). This step is delineated in the flowchart in Fig. 1 between the node "Begin" and the first gray node, it is also presented in detail in Sect. 5.3.

Second, the makespan deviation is assessed for all $\Delta N^y \geq a_0$. This is done through p successive iterations (where p is a nonnegative integer that will be shown to be less than m). These iterations are delineated in the flowchart, from the node "$p = p + 1$" to the node "End", and are addressed in Sect. 5.4.

5.2.2 Illustration

The proposed approach is illustrated through the following example. How the results are produced is shown in Sects. 5.3 and 5.4.

$$Q' = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \qquad P' = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad V' = \begin{bmatrix} 1 & 1 \end{bmatrix}, \qquad N' = \begin{bmatrix} 6 \\ 10 \end{bmatrix}$$

Solving *LP* leads to:

$$R' = \begin{bmatrix} 6 & 0 \\ 2 & 8 \end{bmatrix}, \qquad C_{\max}(Q', N') = 8$$

The completion time deviation is now investigated in the case where the quantity of products of type $y = 1$ increases. Since R' is balanced, $I^c = I$, $J^c = J$ and $a_0 = 0$: indeed, it is not possible to increase the quantity of products of type 1 while keeping the makespan value equal to 8 units of time.

It can be observed that the products of type 1 can only be processed by machine 1. At first glance, it can be thought that the makespan deviation is of one unit of time for each additional product of type 1. However, this is not the case because machine 2 "helps" machine 1 by processing more products of type 2. This phenomenon can be observed for $\Delta N^1 = 1$ as shown below:

$$N'' = N' + \Delta N^1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 7 \\ 10 \end{bmatrix}, \qquad R'' = \begin{bmatrix} 7 & 0 \\ 1.5 & 8.5 \end{bmatrix}, \qquad C_{\max}(Q', N'') = 8.5$$

Thus, the makespan is incremented by 0.5 while the additional quantity of products of type 1 is incremented by 1. However, this increase rate is valid only up to $\Delta N^1 = 4$:

$$N''' = N' + 4 \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \qquad R''' = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, \qquad C_{\max}(Q', N''') = 10$$

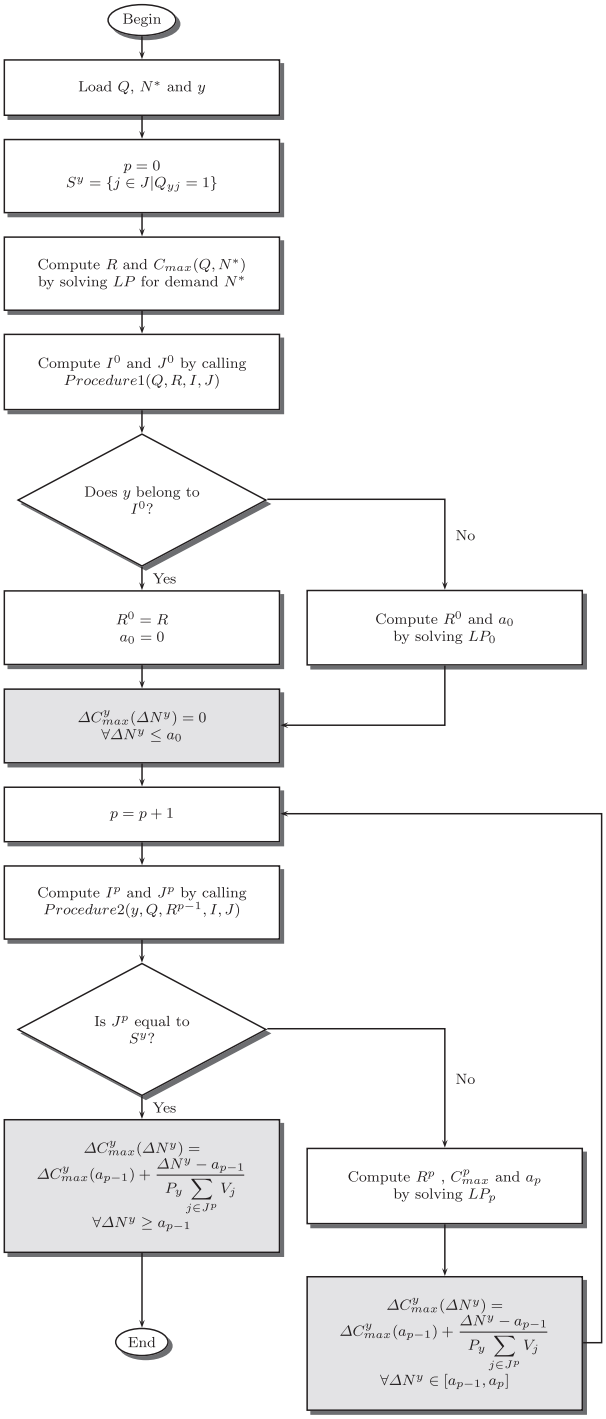


Fig. 1 Approach overview for a single product type

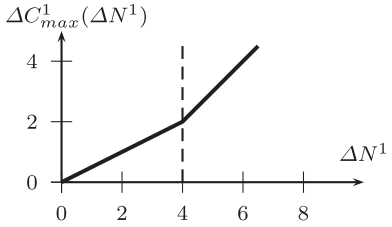


Fig. 2 The completion time deviation for $y = 1$

As a matter of fact, machine 2 cannot help machine 1 cope with a new increase of ΔN^1 because machine 1 is no longer processing any product of type 2. Then, the makespan increases at the same rate as ΔN^1 provided that $\Delta N^1 \geq 4$.

These results are illustrated in Fig. 2 where the makespan deviation $\Delta C^1_{\max}$ is plotted as a function of ΔN^1.

It is noticeable that there are two "times" in the makespan deviation. Each iteration in the lower loop of the flowchart aims at determining the slope of a segment (i.e. the machines that can cope with an increase of the quantity of product of type y), as well as the corresponding values for ΔN^1:

$$\begin{cases} \Delta C^1_{\max}(\Delta N^1) = \frac{\Delta N^1}{2} & (\forall \Delta N^1 \in [0,4]) \\ \Delta C^1_{\max}(\Delta N^1) = 2 + \frac{\Delta N^1 - 4}{1} & (\forall \Delta N^1 \geq 4) \end{cases}$$

The next sections provide the theoretical material and the method used for getting such results.

5.3 Step 0: Finding the maximum value for ΔN^y with no makespan deviation

5.3.1 A linear program for finding the maximum value for ΔN^y with no makespan deviation

First, the maximum quantity of products of type y that can be added to the forecast demand N^* without passing $C_{\max}(Q, N^*)$ is denoted by a_0, and computed by solving LP_0:

$$(LP_0): \begin{cases} \text{Maximize } a_0 & (1) \\ \sum_{j\in J} Q_{ij} P_i V_j R^0_{ij} = N^*_i \quad (\forall i \in I, i \neq y) & (2) \\ \sum_{j\in J} Q_{yj} P_y V_j R^0_{yj} = N^*_y + a_0 & (2') \\ \sum_{i\in I} Q_{ij} R^0_{ij} \leq C_{\max}(Q, N^*) \quad (\forall j \in J) & (3) \\ R^0_{ij} \geq 0 \quad (\forall i \in I)\ (\forall j \in J) & (4) \end{cases}$$

The decision variables in LP_0 are a_0 and R^0_{ij} for all (i, j) in $I \times J$. Since $C^*_{\max}(Q, N^*)$ is not a decision variable, it must be stressed that LP must be previously solved before solving LP_0.

It can be noticed that if $y \in I^c$ (i.e. if y belongs to the product types processed by the critical machines in R), then the optimal value for a_0 is necessarily zero. This shows that in practice, computing I^c with the *Procedure*1 after solving LP can save time because it is not necessary to solve LP_0 if y belongs to I^c.

After performing step 0, it can be deduced that

$$\Delta C^{y}_{\max}(\Delta N^{y}) = 0 \quad \left(\forall \Delta N^{y} \in [0, a_0]\right)$$

5.3.2 Example

In the rest of this paper, the configuration and speed matrices are those defined in Sect. 3.1.2. The forecast demand N^* is defined by $N^* = N^a$, where N^a is also defined in Sect. 3.1.2. Invoking *Procedure*1(Q, R^a, I, J) returns $I^c = \{3, 4, 5\}$. Then, solving LP_0 is necessary for $y = 1$ and $y = 2$ only. It turns out that $a_0 = 2$ for $y = 1$ and $y = 2$. That means that up to two units of products of type 1 (or two units of products of type 2) can be added to N^a with no impact on the makespan value.

5.4 Step $p \geq 1$: An interval approach for assessing $\Delta C^{y}_{\max}(\Delta N^{y})$

This section aims at determining $\Delta C^{y}_{\max}(\Delta N^{y})$ for all $\Delta N^{y} > a_0$. It will be shown that $\Delta C^{y}_{\max}$ is a piecewise linear function as the makespan deviation is linear for all ΔN^{y} in the interval $[a_{p-1}, a_p]$. For any $p \geq 1$, the value for a_p is the objective value returned after solving the linear program LP_p shown in Sect. 5.4.1.

R^{p-1} and a_{p-1} are known after solving LP_{p-1}. Step $p = 1$ starts with the computation of I^p and J^p by invoking *Procedure*2(y, Q, R^{p-1}, I, J), where $I^0 = I^c$ and $J^0 = J^c$ as shown in the flowchart in Fig. 1. The set J^p is the set of machines that can be involved in processing an additional quantity of product of type y. The product types in I^p are the ones that these machines are processing. By construction of *Procedure*2, y always belongs to I^p. Moreover, the set of the qualified machines for y, denoted by S^y is included in J^p for the same reason.

As shown in the lower part of the flowchart in Fig. 1, two situations can occur after computing I^p and J^p. They are delineated in the four following subsections, along with examples.

5.4.1 Case 1: $J^p \neq S^y$

If $J^p \neq S^y$, it must be noticed that the makespan value only depends on the product types in I^p and on the machines in J^p. Consequently, scheduling the type of products in $I^c \backslash I^p$ on the machines in $J^c \backslash J^p$ is an independent subproblem with no impact on the makespan value of the original problem. Then, the production plan R^{p-1} restricted to the product types in I^{p-1} and the machines in J^{p-1} is balanced, and I^p and J^p are computed by calling *Procedure*2(y, Q, R^{p-1}, I, J).

The linear program denoted by (LP_p) aims at finding a production plan R^p such that the additional quantity of products of type y, denoted by a_p, is maximum.

$$(LP_p): \begin{cases} \text{Maximize } a_p & (1) \\ \sum_{j \in J^p} Q_{ij} P_i V_j R^p_{ij} = N^*_i \quad (\forall i \in I^p, i \neq y) & (2) \\ \sum_{j \in J^p} Q_{yj} P_y V_j R^p_{yj} = N^*_y + a_p & (2') \\ \sum_{i \in I^p} Q_{ij} R^p_{ij} = C^p_{\max} \quad (\forall j \in J^p) & (3) \\ R^p_{ij} \geq 0 \quad (\forall i \in I^p)\ (\forall j \in J^p) & (4) \end{cases}$$

This linear program is defined on I^p and J^p since the load of the machines in $J^c \backslash J^p$ depends on the product types in $I^c \backslash I^p$, and is already known from $R^{p'}$, for all $p' \in [0, p-1]$. The set of constraints (3) ensures that all the machines in J^p share the same load ($C^p_{\max}$ is a decision variable). This constraint is mandatory because R^p has to be an optimal solution to LP, and it also has to be balanced since the completion time value must depend on all the machines in J^p and on all the product types in I^p.

Lemma 9 *R^p is r_k-split for $p \geq 1$.*

Proof Let p be greater than or equal to one. By construction, R^p is a balanced production plan, and is an optimal solution to LP_p. If it was not r_k-split, then the completion time value would be a function of all the machines and all the product types J^p and I^p (as can be deduced from the contraposition of Lemma 8). Consequently, it would have been possible to increase strictly a_p while maintaining the balance property, which is a contradiction. □

Then, it can be deduced from Lemma 9 that the machines in J^p are strictly included in the machines in J^{p-1} since some critical machines for R^{p-1} are no longer critical when the amount of products of type y increases (i.e. for R^p). This shows that $p \leq m$, so the approach for determining the completion time deviation for any product type y requires at most m iterations, where such an iteration consists in calling *Procedure*2 and solving LP_p as shown in Fig. 1.

The completion time deviation is now shown to be a non decreasing piecewise linear function as follows. For all ΔN^y in $[a_{p-1}, a_p]$, the makespan can be written as

$$C_{\max}(Q, N^y) = C_{\max}(Q, N^* + a_{p-1}e^y) + \frac{\Delta N^y - a_{p-1}}{P_y \sum_{j \in J^p} V_j}$$

where e^y is the n-by-1 column vector defined by $e^y_y = 1$ and $e^y_i = 0$ for all $i \neq y$.

Then, the makespan deviation for all ΔN^y in $[a_{p-1}, a_p]$ is:

$$\Delta C^y_{\max}(\Delta N^y) = C^{p-1}_{\max} - C_{\max}(Q, N^*) + \frac{\Delta N^y - a_{p-1}}{P_y \sum_{j \in J^p} V_j}$$

$$\Delta C^y_{\max}(\Delta N^y) = \Delta C^y_{\max}(a_{p-1}) + \frac{\Delta N^y - a_{p-1}}{P_y \sum_{j \in J^p} V_j}$$

This shows that the makespan deviation is a non decreasing piecewise linear function of ΔN^y because:

- $\Delta C^y_{\max}$ is an affine function of ΔN^y in $[a_{p-1}, a_p]$ for all p
- The derivative of $\Delta C^y_{\max}$ is $\frac{1}{P_y \sum_{j \in J^p} V_j}$ which is a positive quantity
- $\Delta C^y_{\max}$ is continuous at $\Delta N^y = a_p$ for all p as $\Delta C^y_{\max}(a_p) = \Delta C^y_{\max}(a_{p-1}) + \frac{a_p - a_{p-1}}{P_y \sum_{j \in J^p} V_j}$.

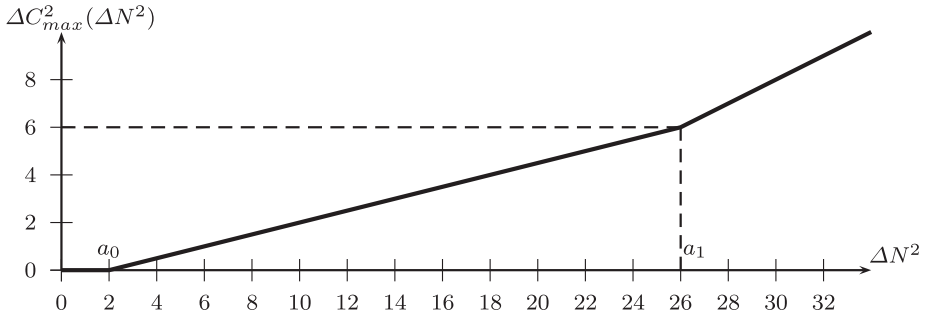


Fig. 3 The completion time deviation for $y = 2$

5.4.2 Example

Let us consider the product type $y = 2$. It has been shown in Sect. 5.3.2 that $a_0 = 2$, then R^0 is given below:

$$R^0 = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 2 & 0 & 8 \\ 0 & 0 & 12 & 4 \end{bmatrix}$$

Since R^0 is balanced, all the machines and all the product types are critical. Furthermore, it is r_k-split, and invoking *Procedure*2$(2, Q, R^0, I, J)$ leads to $I^1 = I$ and $J^1 = J$. Solving LP_1 yields:

$$R^1 = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 13 & 18 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \\ 0 & 0 & 8 & 8 \end{bmatrix}, \qquad a_1 = 26, \qquad C^1_{\max} = 18$$

Then the completion time deviation is zero for all ΔN^2 in $[0, 2]$ and:

$$\Delta C^2_{\max}(\Delta N^2) = 0 + \frac{\Delta N^2 - 2}{P_2 \sum_{j \in J^1} V_j} = \frac{\Delta N^2 - 2}{4} \quad (\forall \Delta N^2 \in [2, 26])$$

Since $J^1 \neq S^2$, *Procedure*2 must be invoked with production plan R^1 as follows: *Procedure*2$(2, Q, R^1, I, J)$. It yields $I^2 = \{1, 2\}$ and $J^2 = \{1, 2\}$. Thus, $J^2 = S^2$. The completion time deviation in that case is studied in the next section. The results found up to this point (i.e. $\Delta C^2_{\max}(\Delta N^2)$ for $\Delta N^2 \in [0, 26]$) are shown in Fig. 3.

5.4.3 Case 2: $J^p = S^y$

If $J^p = S^y$ then these machines remain critical for any additional quantity of products of type y.

By definition of LP_{p-1}, the demand defined by $N^* + a_{p-1}.e^y$ has an optimal completion time denoted by $C^{p-1}_{\max}$. Consequently, for all $\Delta N^y \geq a_{p-1}$, the completion time $C_{\max}(Q, N^y)$

is defined by:

$$C_{\max}(Q, N^y) = C_{\max}^{p-1} + \frac{\Delta N^y - a_{p-1}}{P_y \sum_{j \in J^p} V_j}$$

$$C_{\max}(Q, N^y) = C_{\max}(Q, N^* + a_{p-1} e^y) + \frac{\Delta N^y - a_{p-1}}{P_y \sum_{j \in J^p} V_j}$$

Then, for all $\Delta N^y \geq a_{p-1}$, the makespan deviation is:

$$\Delta C_{\max}^y(\Delta N^y) = \Delta C_{\max}^y(a_{p-1}) + \frac{\Delta N^y - a_{p-1}}{P_y \sum_{j \in J^p} V_j}$$

5.4.4 Example

It has been shown in Sect. 5.4.2 that from $\Delta N^2 \geq 26$, $J^2 = S^2$. In other words, the only machines that can be used to cope with an amount of product of type 2 that is more than 26 are the machines that are qualified for product type 2. With $p = 2$, $a_{p-1} = 26$ and $\Delta C_{\max}^2(a_{p-1}) = 6$. Consequently, it can be deduced that:

$$\Delta C_{\max}^2(\Delta N^2) = 6 + \frac{\Delta N^2 - a_1}{P_2 \sum_{j \in J^2} V_j} = 6 + \frac{\Delta N^2 - 26}{2} \quad (\forall \Delta N^2 \geq 26)$$

The completion time deviation can be plotted for all nonnegative ΔN^2 as shown in Fig. 3.

5.5 Computational complexity analysis of the overall approach

The computational complexity analysis of the overall approach presented in this section relies on the analysis of its subprocesses as shown in Fig. 1. Since this approach involves linear programs LP, LP_0 and LP_p, it is necessary to assess the computational complexity of solving each of them. It can be found in Wright (1997) that the best known complexity bound for solving a linear program having n' variables, m' linearly independent constraints and rational entries is $\mathcal{O}(L\sqrt{n'})$ for the interior-point algorithm presented in Renegar (1988), where L is the total length of the data string (i.e. the problem size). LP has $n' = nm$ variables and $m' = n + m$ constraints, LP_0 and LP_p have $n' = nm + 1$ variables and $m' = n + m$ constraints for all p. As a consequence, the computational complexity of solving any of these linear programs is $\mathcal{O}(Ln)$.

The steps that lie between "Begin" and "$p = p + 1$" in Fig. 1 are executed only once. LP (and possibly LP_0) are solved, and *Procedure*1 is called once, so the complexity of this part of the approach is $\mathcal{O}(n^3)$.

It can be seen that the loop involving the call to *Procedure*2 and the solution of LP_p is repeated until $J^p = S^y$, with initially $p = 1$. It has been shown after proving Lemma 9 that $p \leq m$, thus the complexity of this part of the approach is $\mathcal{O}(n^4)$.

Finally, the global complexity of the proposed approach for assessing the completion time deviation for a single product type is $\mathcal{O}(n^4)$.

6 Sensitivity analysis for all the product types

In this section, it is shown that the completion time deviation $\Delta C_{\max}$ resulting from an additive quantity of unexpected demand ΔN having a magnitude equal to a can be upper

bounded by the maximum deviation resulting from an additive quantity of unexpected demand on a single product type having the same magnitude. More formally, the following theorem holds:

Theorem 1 *Let ΔN be a nonnegative column vector modelling demand uncertainty, and its magnitude a be defined by $a = \|\Delta N\|_1$. Then the following inequality holds*:

$$\Delta C_{\max}(a) \leq \max_{i \in I} \left\{ \Delta C^i_{\max}(a) \right\}$$

The L1-norm is used for measuring demand uncertainty because $\|\Delta N\|_1$ returns the total number of products that are added to the forecast demand N^*.

Proof For any nonnegative vector ΔN modelling demand uncertainty, it can be written:

$$\Delta N = \|\Delta N\|_1 . \sum_{i \in I} \left(\alpha_i . e^i \right)$$

where α_i is such that $0 \leq \alpha_i \leq 1$ for all i in I, and $\sum_{i \in I} \alpha_i = 1$.

Then,

$$N^* + \Delta N = \left(\sum_{i \in I} \alpha_i \right) . N^* + a . \left(\sum_{i \in I} \alpha_i . e^i \right)$$

$$N^* + \Delta N = \sum_{i \in I} \left(\alpha_i . \left(N^* + a . e^i \right) \right)$$

It has been shown in Aubry et al. (2008b) that $C_{\max}(Q, N)$ satisfies the triangular inequality on N, i.e. if $N = N^\alpha + N^\beta$ then $C_{\max}(Q, N) \leq C_{\max}(Q, N^\alpha) + C_{\max}(Q, N^\beta)$. Hence:

$$C_{\max}(Q, N^* + \Delta N) \leq \sum_{i \in I} \left(\alpha_i . C_{\max}(Q, N^* + a . e^i) \right) \leq \left(\sum_{i \in I} \alpha_i \right) . \max_{i \in I} \left\{ C_{\max}(Q, N^* + a . e^i) \right\}$$

And then:

$$\Delta C_{\max}(a) \leq \max_{i \in I} \left\{ \Delta C^i_{\max}(a) \right\} \qquad \square$$

It can be checked that the upper bound provided for $\Delta C_{\max}(a)$ is the best possible one as it is reached for all a by the vector $\Delta N = a . e^y$ modelling demand uncertainty, where y is defined by $\Delta C^y_{\max}(a) = \max_{i \in I} \{ \Delta C^i_{\max}(a) \}$.

As a consequence of Theorem 1, the results shown in the previous section can easily be used to assess the completion time deviation $\Delta C_{\max}$ resulting from any additive unexpected demand having a magnitude that is less than or equal to a.

$$\Delta C_{\max}(a) = \max_{i \in I} \left\{ \Delta C^i_{\max}(a) \right\} \quad \forall a \geq 0$$

6.1 Example

The sensitivity analysis for product type 2 has been performed in Sects. 5.4.2 and 5.4.4. It is done the same way for the remaining product types. Then, $\Delta C^y_{\max}$ is plotted as a function

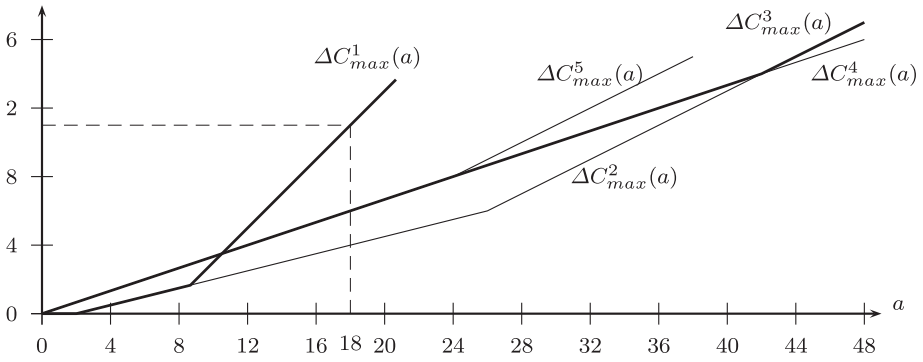


Fig. 4 Completion time deviation with Q and N^* for each single product type as a function of a

of a for all y in Fig. 4, where $a = \Delta N^y$ for each single product type y. For the sake of readability, $\Delta C^1_{\max}$ and $\Delta C^3_{\max}$ have been plotted with solid lines.

Finally, $\Delta C_{\max}(a)$ has the following expression:

$$\begin{cases} \Delta C_{\max}(a) = \frac{a}{3} & (\forall a \in [0, \frac{21}{2}]) \\ \Delta C_{\max}(a) = \frac{7}{2} + \frac{a - \frac{21}{2}}{1} & (\forall a \geq \frac{21}{2}) \end{cases}$$

This shows that when $a = 18$, the makespan deviation is less than 11 units of time. As can also be seen in Fig. 4, it can be observed that product type 1 is responsible for this result. This indicates that the completion time deviation is exactly 11 units of time if the entire demand increase is focused on the products of type 1.

Such results are meaningful as they let the decision-maker know which type of demand uncertainty may have the greatest impact on the makespan for configuration Q. Different configurations can be compared on the basis of completion time deviation.

6.1.1 A very particular case

When there are no qualification constraints, the workshop can be viewed as a set of m uniform machines and the configuration matrix is defined by $Q^*_{ij} = 1$ for all (i, j) in $I \times J$. In that case, the completion time deviation is of the following form:

$$\Delta C^y_{\max}(\Delta N^y) = \frac{\Delta N^y}{P_y \sum_{j \in J} V_j} \quad (\forall \Delta N^y \geq 0)\ (\forall y \in I)$$

Indeed, since any optimal production plan for Q^* is balanced for all N, a_0 is zero for all y in I, and because all the machines are qualified for all the product types, $J^1 = S^1$ for all y in I. In such a case, all the machines can always be used for dealing with the increase of any product type, so the sum over V_j at the denominator of $\Delta C^y_{\max}$ is maximum as it is performed over J. As a consequence, the completion time deviation for all $y \in I$ is minimum over all the configurations.

However, this maximum performance in terms of completion time deviation is achieved at the expense of a maximum configuration cost as all the qualifications are enabled. But even if such a configuration is not feasible for financial or technological reasons, the completion time deviation above can be used as a theoretical lower bound for what can be expected

Table 1 CPU time for performing the sensitivity analysis for all product types

	Config. density	10	20	30	40	50
Max.	0.10	0.08	1.33	12.83	19.14	85.59
	0.25	0.14	2.38	5.83	10.80	24.56
	0.50	0.13	0.78	2.19	6.75	20.43
	0.75	0.05	0.39	1.50	4.32	9.83
Avg.	0.10	0.06	0.83	4.73	13.90	50.80
	0.25	0.09	1.44	3.56	9.49	22.03
	0.50	0.08	0.57	2.14	6.30	17.39
	0.75	0.05	0.36	1.43	4.32	9.40

from any feasible configuration. Furthermore, it also illustrates the fact that the performance of a given configuration can be increased by enabling more qualifications. The next section shows that this simple (but costly) strategy is not the only way, as the choice of the qualifications in the configuration matrix also has a major impact on the completion time.

6.1.2 Computational results

The approach presented in this paper is now applied to a large set of instances in order to assess its computational cost. To this end, a configuration matrix Q, a uniform speed matrix W and a forecast demand N^* have been randomly generated with $n = m$ in the set $\{10, 20, 30, 40, 50\}$, and with the configuration matrix density d in $\{0.1, 0.25, 0.5, 0.75\}$. Ten instances have been generated for each couple (n, d), the maximum and average CPU time needed for performing the sensitivity analysis on all product types is expressed in seconds and reported in Table 1. The approach is implemented on a 3.2 GHz Intel Pentium IV Processor system with 1 GBytes of RAM, under Mosel Xpress-MP.

Furthermore, a real-world instance from a photolithography workshop with $n = 36$ product types to process, $m = 13$ machines, and whose density is 0.24, has been solved in less than 4 seconds (data can be found in Rossi 2003). As can be seen in Table 1, besides instance size, the configuration matrix density has a drastic impact on the computational effort required for executing the proposed approach. This is due to the fact that many more rectangles of zeros are likely to be found in a sparse configuration matrix, leading to more r_k-split production plans. Consequently, the value for p is higher and so is the number of iterations of the loop in the lower part of the flowchart in Fig. 1. However, even for low-density configurations, the approach requires a reasonably short amount of CPU time up to 50 machines and 50 product types, which is beyond the size of a real-life photolithography workshop to the authors' best knowledge.

6.2 Comparing configurations on the basis of completion time deviation

The decision-maker, who is responsible for designing a robust configuration for the workshop, can use the results presented in this paper as they provide an insight of the performance deviation when the demand is subject to uncertainty. For illustrating this point, a new con-

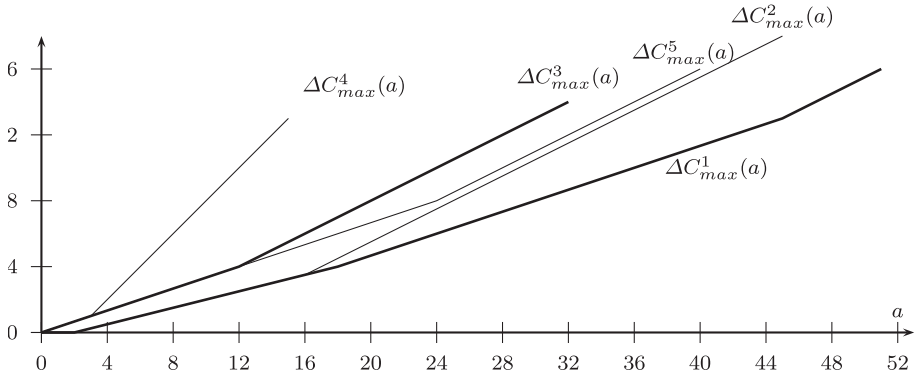


Fig. 5 Completion time deviation with Q' and N^* for each single product type as a function of a

figuration Q' is compared with the original one introduced in Sect. 3.1.2.

$$Q' = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

First, it can be observed that the cost of configuration Q' is 9 (with unitary costs), which is one less than the cost of Q. Solving *LP* leads to a production plan for $N^* = N^a$ (where N^a is defined in Sect. 3.1.2) whose makespan is $C_{\max}(Q', N^*) = 12$ units of time, which is the same value as with configuration Q.

The completion time deviation is assessed for each single product type for Q' and N^*, the results are shown in Fig. 5.

As can be seen in Fig. 5, the maximum completion time deviation is set by product type 4, its expression is:

$$\begin{cases} \Delta C_{\max}(a) = \Delta C^4_{\max}(a) = \frac{a}{3} & (\forall a \in [0, 3]) \\ \Delta C_{\max}(a) = \Delta C^4_{\max}(a) = a - 2 & (\forall a \geq 3) \end{cases}$$

Thus, when comparing Q and Q' on the basis of the maximum completion time deviation as a function of the magnitude of demand uncertainty, it can be observed that they behave the same way up to $a = 3$. Beyond that threshold, the makespan increases faster with Q' than with Q as configuration Q can maintain a lower makespan increase rate up to $a = 10.5$. As $a = 3$ is around 6% of the total volume of N^*, it can be said that in a lowly-perturbed context (i.e. when the magnitude of unexpected demands does not exceeds 6% of the forecast demand volume), Q offers no advantage over Q'. However, the results of Sect. 6.1.1 show that better results could be achieved with a different configuration, as the makespan deviation may, at least to some extend, be as low as $a/4$.

Now we consider a much more perturbed environment as we assume that the magnitude of unexpected demands can be up to $a = 18$ (which represents nearly 40% of the volume of N^*). Table 2 displays the completion time deviation achieved for Q and Q', for an 18-magnitude additional demand on every single product type.

Table 2 Completion time deviation comparison for an 18-magnitude additional demand

Configuration	$\Delta C^1_{\max}$	$\Delta C^2_{\max}$	$\Delta C^3_{\max}$	$\Delta C^4_{\max}$	$\Delta C^5_{\max}$
Q	11	4	6	6	6
Q'	4	$\frac{9}{2}$	7	16	6

It can be seen in Table 2 that configuration Q clearly outperforms Q' as the completion time deviation is at most 11 for Q and at most 16 for Q'. But this may not lead to systematically favor Q over Q', as configuration Q is less costly than Q, so the decision-maker should be aware that the configuration cost can decrease by 1 at the expense of an increase by 5 of the maximum completion time deviation if Q is to be replaced by Q'.

The results presented in this paper can be used for carrying out deeper analyses: if the decision-maker's experience suggests that the volume of the products of type 4 is very unlikely to increase during the lifetime of the configuration, the interpretation of the sensitivity analysis may lead to drawing different conclusions on Q and Q'. Indeed, if we assume that no additional demand of products of type 4 is possible, the maximum completion time deviation for an 18-magnitude additional demand remains unchanged for Q, and drops to 7 for Q' (see Table 2). Consequently, under such a hypothesis, Q' appears to be significantly more advantageous than Q as it is less costly and offers better performance.

The following briefly illustrates the use of the proposed sensitivity analysis when a robust solution is to be found. Following the robustness framework presented in Billaut et al. (2008), the set of instances $\mathcal{P}$ over which a robust configuration is searched for must be carefully specified as it measures the risk that is to be covered by robust configurations. For example, if the decision-maker wants a configuration to be robust for all the demands such that the magnitude of additive perturbations can be up to 18, $\mathcal{P}$ must be defined as:

$$\mathcal{P} = \{N \in \mathbb{R}^n_+ | N \leq N^* + \Delta N, \|\Delta N\|_1 = 18\}$$

It should be noticed that $\mathcal{P}$ includes additive and non-additive perturbations on N^* with no inconvenience, because non-additive perturbations cannot increase the makespan (see Aubry et al. 2008b).

If the absolute robustness is chosen, the robustness criterion is written as:

$$\lambda_1 = \max_{N \in \mathcal{P}} \big(C_{\max}(Q, N)\big)$$

Then, configuration Q is absolutely robust for the set of instances $\mathcal{P}$ over the configurations set $\{Q, Q'\}$ and Q' is not.

If the $\mathcal{L}_\lambda$-robustness is used with $\lambda = \lambda_1$, then the value for $\mathcal{L}_\lambda$ can be set in such a way that $C_{\max}(Q^*, N^*) + \mathcal{L}_\lambda = D$, where D is the due date for the demand. Thus, a configuration is said to be robust if the due date can be met for all demands in $\mathcal{P}$. For example, if $D = 30$, then $\mathcal{L}_\lambda = 18$ and configurations Q and Q' are robust.

As a conclusion to this section, it has been shown that the sensitivity analysis carried out in this article is useful for assessing the behavior of a given configuration when the demand is subject to perturbations, and it also provides a bound on the best achievable performance by considering configuration Q^*. However, sensitivity analysis alone is not sufficient for making appropriate decisions on the design of the configuration, as designing or updating a configuration involves many other aspects (like configuration cost and qualification constraints) that go beyond the scope of makespan deviation assessment.

7 Conclusion

This paper shows that it is possible to assess the completion time deviation in a multi-purpose machine workshop facing demand uncertainty. The proposed approach is polynomial as it relies on n sensitivity analyses whose complexity is $\mathcal{O}(n^4)$. This study is for uniform machines only, as can be seen this assumption is widely used in the theoretical part of this work.

The proposed results can be used in a decision-aid tool, in order to help a decision-maker with the configuration of a multi-purpose machine workshop. Thus, knowing the configuration performance when unexpected demands occur allows one to compare configurations on the basis of their performance under uncertainty, and to make informed choices on the configuration. Moreover, this work can also be useful when managing the workshop on-line as it allows for assessing production lateness with accuracy, this piece of information being valuable when it comes to negotiating delays with customers. However, as shown in the last section, sensitivity analysis alone is not sufficient for properly designing or updating a configuration as it only captures a single aspect of a complex problem. In other words, the proposed results in their current form are intended to be suitable for decision-aiding, but not for automated decision-making.

Hereafter, this work should be extended for providing the decision-maker with strategies to build configurations from scratch. As qualifying machines is often very costly and time-consuming, the provided help should contribute to identifying a trade-off between the configuration cost and its ability to deal with demand uncertainty.

References

Akçali, E., Nemoto, K., & Uzsoy, R. (2001). Cycle-time improvements for photolithography process in semiconductor manufacturing. *IEEE Transaction on Semiconductor Manufacturing*, *14*(1), 48–56.

Aubry, A., Rossi, A., & Jacomino, M. (2008a). Minimizing setup costs for parallel multi-purpose machines under load-balancing constraint. *European Journal of Operational Research*, *187*(3), 1115–1125.

Aubry, A., Rossi, A., & Jacomino, M. (2008b). Sensitivity analysis for the configuration of a multi-purpose machines workshop. In *International federation of automatic control*, Seoul, South Korea. Available at http://sites.google.com/site/aubryalexis/.

Aubry, A., Rossi, A., & Jacomino, M. (2009). A generic off-line approach for dealing with uncertainty in optimisation. In *Preprints of the 13th IFAC symposium on information control problems in manufacturing*, Moscow, Russia, 3–5 June (pp. 1464–1469).

Brucker, P., Jurisch, B., & Kramer, A. (1997). Complexity of scheduling problems with multi-purpose machines. *Annals of Operational Research*, *70*, 57–73.

Billaut, J.-C., Moukrim, A., & Sanlaville, E. (2008). *Flexibility and robustness in scheduling*. London: ISTE Ltd, Wiley.

Espinouse, M.-L., Jacomino, M., & Rossi, A. (2008). The robustness of multi-purpose machines workshop configuration. In *Flexibility and robustness in scheduling*. London: ISTE Ltd, Wiley.

Graham, R., Lawler, E., Lenstra, J., & Rinnooy Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, *5*, 287–326.

Kouvelis, P., & Yu, G. (1997). *Robust discrete optimization and its applications*. Dordrecht: Kluwer Academic.

Kouvelis, P., Kurawarwala, A. A., & Gutiérrez, G. J. (1992). Algorithms for robust single and multiple period layout planning for manufacturing systems. *European Journal of Operational Research*, *63*, 287–303.

Lawler, E., & Labetoulle, J. (1978). On preemptive scheduling of unrelated parallel processors by linear programming. *Journal of the Association for Computing Machinery*, *25*, 612–619.

Leung, J., & Li, C.-L. (2008). Scheduling with processing set restrictions: a survey. *International Journal of Production Economics*, *116*, 251–262.

Nourine, L., & Raynaud, O. (1999). A fast algorithm for building lattices. *Information Processing Letters*, *71*(5–6), 199–204.

Penz, B., Rapine, C., & Trystram, D. (2001). Sensitivity analysis of scheduling algorithms. *European Journal of Operational Research*, *134*, 606–615.
Renegar, J. (1988). A polynomial-time algorithm, based on Newton's method, for linear programming. *Mathematical Programming*, *40*, 59–93.
Rossi, A. (2003). *Ordonnancement en milieu incertain, mise en œuvre d'une démarche robuste*. Ph.D. thesis, Institut National Polytechnique de Grenoble, France.
Snyder, L. V. (2006). Facility location under uncertainty: a review. *IIE Transactions*, *38*(7), 547–564.
Wright, S. (1997). *Primal-dual interior-point methods*. Philadelphia: Society for Industrial and Applied Mathematics.
Yoon, H. J., & Lee, D. Y. (2004). Deadlock-free scheduling of photolithography equipment in semiconductor fabrication. *IEEE Transaction on Semiconductor Manufacturing*, *17*(1), 42–54.

Discrete Applied Mathematics 159 (2011) 2281–2289

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Note

Three new upper bounds on the chromatic number

María Soto *, André Rossi, Marc Sevaux

Université de Bretagne-Sud, Lab-STICC, CNRS UMR 3192, Centre de Recherche B.P. 92116, F-56321 Lorient Cedex, France

ARTICLE INFO

Article history:
Received 9 July 2010
Received in revised form 16 June 2011
Accepted 1 August 2011
Available online 23 September 2011

Keywords:
Graph coloring
Chromatic number
Upper bounding scheme

ABSTRACT

This paper introduces three new upper bounds on the chromatic number, without making any assumptions on the graph structure. The first one, ξ, is based on the number of edges and nodes, and is to be applied to any connected component of the graph, whereas ζ and η are based on the degree of the nodes in the graph. The computation complexity of the three-bound computation is assessed. Theoretical and computational comparisons are also made with five well-known bounds from the literature, which demonstrate the superiority of the new upper bounds.

1. Introduction

An undirected, simple graph is given; the graph coloring problem is to assign a color to every node in such a way that two adjacent nodes do not have the same color, while minimizing the total number of colors used. This problem arises in many practical applications, such as map coloring, timetabling, scheduling, memory allocation, and many others [6]. Formally, a coloring of graph $G = (X, U)$ is a function $F : X \to \mathbb{N}^*$, where each node in X is allocated an integer value that is called a color. A proper coloring satisfies $F(u) \neq F(v)$ for all $(u, v) \in U$ [4,7]. A graph is said to be α-colorable if there exists a coloring which uses, at most, α different colors. In that case, all the nodes colored with the same color are said to be part of the same class. The smallest number of colors involved in any proper coloring of a graph G is called the *chromatic number*, which is denoted by $\chi(G)$. The problem of finding $\chi(G)$, as well as a minimum coloring, is $\mathcal{NP}$-hard and is still the focus of an intense research effort [2,3,9,10].

First, we recall some elementary results on the graph coloring problem, and introduce some notations. A graph cannot be α-colorable with $\alpha < \chi(G)$. The chromatic number equals 1, if and only if G is a totally disconnected graph. It is equal to $|X|$ if G is complete, and for the graphs that are exactly bipartite (including trees and forests), the chromatic number is 2.

Let G be a non directed, simple graph, where $n = |X|$ is the number of nodes, and $m = |U|$ is the number of edges. The degree of node i is denoted by d_i for all $i \in \{1, \ldots, n\}$, and $\delta(G)$ is the highest degree in G. The following upper bounds on $\chi(G)$ can be found in the literature:

- $\chi(G) \leq d = \delta(G) + 1$ [4,6].
- $\chi(G) \leq l = \left\lfloor \frac{1+\sqrt{8m+1}}{2} \right\rfloor$ [4,6].
- $\chi(G) \leq M = \max_{i \in X} \min(d_i + 1, i)$, provided that $d_1 \geq d_2 \geq \cdots \geq d_n$ [13].
- $\chi(G) \leq s = \delta_2(G) + 1$, where $\delta_2(G)$ is the largest degree that a node v can have, if v is adjacent to a node whose degree is at least as large as its own [11].
- $\chi(G) \leq q = \left\lceil \frac{r}{r+1}(\delta(G) + 1) \right\rceil$, where r is the maximum number of nodes of the same degree, each at least $(\delta(G) + 2)/2$ [12].

* Corresponding author. Tel.: +33 0 297874562; fax: +33 0 297874527.
E-mail addresses: maria.soto@univ-ubs.fr (M. Soto), andre.rossi@univ-ubs.fr (A. Rossi), marc.sevaux@univ-ubs.fr (M. Sevaux).

0166-218X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2011.08.005

Regarding lower bounds, the chromatic number is greater than or equal to the clique number denoted by $\omega(G)$, which is the size of the largest clique in the graph; thus $\omega(G) \leq \chi(G)$. However, this bound is difficult to use in practice as finding the clique number is $\mathcal{NP}$-hard, and the Lovasz number is known to be a better lower bound for $\chi(G)$ as it is "sandwiched" between the clique number and the chromatic number [8]. Moreover, the Lovasz number can be calculated in polynomial time.

There exist some upper bounds on the chromatic number for special classes of graphs:

- $\chi(G) \leq \delta(G)$, for a connected, simple graph which is neither complete, nor has an odd cycle.
- $\chi(G) \leq 4$, for any planar graph.

In Section 2, three new upper bounds on the chromatic number are proposed. The quality of these bounds is then compared with existing bounds in Section 3, and computational experiments are conducted in Section 4 for assessing the practical improvement of the three new upper bounds.

2. Three new upper bounds on the chromatic number

The following lemma is required for proving Theorem 1, which introduces the first bound proposed in this paper.

Lemma 1. *The following inequality holds for any connected, simple graph $G_n = (V, E)$, where $m_n = |E|$.*

$$\frac{\chi(G_n)\,(\chi(G_n)-1)}{2} + n - \chi(G_n) \leq m_n. \tag{1}$$

This inequality is referred to as Eq. (1).

Proof. Lemma 1 is proved by recurrence on n.

First, it can be observed that Lemma 1 is obviously true for $n = 2$. Indeed, there exists a unique connected, simple graph on two vertices which has a single edge, and $\chi(G_2) = 2$.

Second, we assume that Lemma 1 is valid for all graphs having at most n vertices. We now prove that the inequality Lemma 1 holds for any connected, simple graphs on $n+1$ vertices. Let such a graph be denoted by G_{n+1}. It has m_{n+1} edges and its chromatic number is $\chi(G_{n+1})$.

G_{n+1} can be seen as a connected, simple graph G_n plus an additional vertex denoted by $n+1$, and additional edges incident to this new vertex. The addition of vertex $n+1$ to G_n either leads to $\chi(G_{n+1}) = \chi(G_n)$, or to $\chi(G_{n+1}) = \chi(G_n) + 1$. Indeed, the introduction of a new vertex (along with its incident edges) into a graph leads to increment of the chromatic number by at most one.

- First case: $\chi(G_{n+1}) = \chi(G_n)$.
  Adding 1 to Eq. (1) yields
  $$\frac{\chi(G_{n+1})\,(\chi(G_{n+1})-1)}{2} + n + 1 - \chi(G_{n+1}) \leq 1 + m_n \leq m_{n+1}.$$
  We have $1 + m_n \leq m_{n+1}$, because at least one new edge is to be added to G_n for building G_{n+1}: vertex $n+1$ has to be connected to at least one edge in G_n for G_{n+1} to be connected.
- Second case: $\chi(G_{n+1}) = \chi(G_n) + 1$.
  A minimal coloring of G_{n+1} can be obtained by keeping the minimal coloring of G_n, and by assigning color $\chi(G_{n+1}) = \chi(G_n) + 1$ to vertex $n+1$. Since this coloring is minimal, there exists at least one edge between any pair of color classes [4]. In particular, this requirement for color $\chi(G_{n+1})$ implies that the degree of vertex $n+1$ is at least $\chi(G_n)$, hence $m_n + \chi(G_n) \leq m_{n+1}$.
  Adding $\chi(G_n)$ to Eq. (1) yields
  $$\left(\frac{\chi(G_n)\,(\chi(G_n)-1)}{2} + \chi(G_n)\right) + n - \chi(G_n) \leq m_n + \chi(G_n).$$
  The quantity in parenthesis is equal to the sum of the integers in $\{1, \ldots, \chi(G_n)\}$, and since $\chi(G_{n+1}) = \chi(G_n) + 1$,
  $$\frac{\chi(G_{n+1})\,(\chi(G_{n+1})-1)}{2} + n - \chi(G_n) \leq m_n + \chi(G_n).$$
  Finally, as $n - \chi(G_n) = n + 1 - \chi(G_{n+1})$ and $m_n + \chi(G_n) \leq m_{n+1}$,
  $$\frac{\chi(G_{n+1})\,(\chi(G_{n+1})-1)}{2} + n + 1 - \chi(G_{n+1}) \leq m_{n+1}. \quad \square$$

Theorem 1. *The following inequality holds for any connected, simple undirected graph G*

$$\chi(G) \leq \xi,$$

with $\xi = \left\lfloor \frac{3+\sqrt{9+8(m-n)}}{2} \right\rfloor$.

Proof. By Lemma 1, m can be lower bounded as follows:

$$\frac{\chi(G)(\chi(G)-1)}{2} + n - \chi(G) \leq m.$$

This inequality leads to the following second order polynomial in the variable $\chi(G)$:

$$\chi(G)^2 - 3\chi(G) - 2(m-n) \leq 0.$$

Once solved, this inequality leads to:

$$\chi(G) \leq \left\lfloor \frac{3+\sqrt{9+8(m-n)}}{2} \right\rfloor. \quad \square$$

Note that, because all connected graphs have at least $n-1$ edges, then $8(m-n)+9 \geq 1$; thus the square root is in $\mathbb{R}^+$.

Remark 1. As this bound is only based on the number of the nodes and edges in the graph, it yields the same value for all graphs having the same number of nodes and edges. This bound computation requires $\mathcal{O}(1)$ operations.

Theorem 2. *For any simple, undirected graph G, $\chi(G) \leq \zeta$, where ζ is the greatest number of nodes with a degree greater than or equal to $\zeta - 1$.*

Theorem 3. *For any simple, undirected graph G, $\chi(G) \leq \eta$, where η is the greatest number of nodes with a degree greater than or equal to η that are adjacent to at least $\eta - 1$ nodes, each of them with a degree larger than or equal to $\eta - 1$.*

Before proving Theorems 2 and 3, some notations and definitions need to be stated. It should be noticed that connectivity is not required for the last two bounds, which involves more information on the graph topology than the first one.

The degree of saturation [1,7] of a node $v \in X$ denoted by $DS(v)$ is the number of different colors of the nodes adjacent to v. For a minimum coloring of graph G, $DS(v)$ is in $\{1, \ldots, \chi(G)-1\}$ for all $v \in X$.

The following notations are used throughout this paper.

- $C = \{1, \ldots, \chi(G)\}$ is the minimum set of colors used in any valid coloring.
- A valid (or proper) coloring using exactly $\chi(G)$ colors is said to be a minimal coloring.
- The neighborhood of node v denoted by $N(v)$ is the set of all nodes u such that edge (u, v) belongs to U. $N(v)$ is also called the set of adjacent nodes to v.

The last two bounds are based on the degree of saturation of a node and on Lemma 2.

Lemma 2. *Let F be a minimal coloring of G. For every color k in C, there exists at least one node v colored with k, (i.e., $F(v) = k$), such that its degree of saturation is $\chi(G) - 1$ and where v is adjacent to at least $\chi(G) - 1$ nodes with a degree larger than or equal to $\chi(G) - 1$.*

Proof of Lemma 2. We prove the lemma by contradiction. First, we show that for all k in C there exists a node v, colored with k, such that $DS(v) = \chi(G) - 1$. To do so, we assume that there exists a color k in C such that any node v colored with k has a degree of saturation that is strictly less than $\chi(G) - 1$.

Then, it can be deduced that for all $v \in X$ such that $F(v) = k$, there exists a color $c \in C \setminus \{k\}$ such that there does not exist $u \in N(v)/F(u) = c$. Consequently, a new valid coloring can be derived from the current one by setting $F(v) = c$. Indeed, v is not connected to any node colored with c. This operation can be performed for any node colored with k, leading to a valid coloring in which color k is never used. Hence, this new coloring involves $\chi(G) - 1$ colors, which is impossible by definition of the chromatic number.

Second, we show that, for every k in C, there exists a node v colored with k, whose degree of saturation is equal to $\chi(G)-1$, and such that v has at least $\chi(G) - 1$ neighbors with degree larger than or equal to $\chi(G) - 1$. To do so, we assume that there exists a color k in C such that any node v colored with k having a degree of saturation equal to $\chi(G) - 1$ has strictly less than $\chi(G) - 1$ neighbors with a degree larger than or equal to $\chi(G) - 1$.

Then, it can be deduced that for all node v colored with k and such that $DS(v) = \chi(G)-1$, there exists one color $c \in C \setminus \{k\}$ such that the degree of any node $w \in V(v)/F(w) = c$ is strictly less than $\chi(G)-1$. Then, for each node $w \in V(v)/F(w) = c$, there exists a color $l \in C \setminus \{k, c\}$ such that setting $F(w)$ to l yields a valid coloring. As a result, color c is no longer used in $N(v)$, thus $DS(v)$ is no longer $\chi(G) - 1$. This operation can be performed for any node v such that $F(v) = k/DS(v) = \chi(G) - 1$, leading to a coloring in which there is no node v colored with k and such that $DS(v) = \chi(G)-1$. It can then be deduced from the first part of this proof that in such a situation, G can be colored with strictly less than $\chi(G)$ colors, which is impossible. $\square$

Proof of Theorem 2. It can be deduced from Lemma 2 that there exists at least $\chi(G)$ nodes in G, with a degree of at least $\chi(G) - 1$. Thus, ζ being the greatest number of nodes with a degree greater than or equal to $\zeta - 1$, the following inequality holds: $\chi(G) \leq \zeta$. $\square$

Remark 2. It can easily be seen that Algorithm 1, which returns ζ, has a computational complexity of $\mathcal{O}(\max\{m, n\log_2(n)\})$, as it requires enumerating the m edges to compute the degree of the nodes, $n\log_2(n)$ operations to sort the nodes, and $\zeta \leq n$ iterations in the `while` loop.

Algorithm 1: Computing ζ.

```
Data: Graph G(X, U); where n ← |X| and m ← |U|.
Compute the degree, d_i of all nodes i in X;
Sort the nodes by non increasing degree;
ζ ← 0, stable ← 0 and i ← 0;
while stable = 0 and i ≤ n do
    if d_i ≥ ζ then
        ζ ← ζ + 1;
    else
        stable ← 1;
    i ← i + 1;
```

Proof of Theorem 3. It can be deduced from Lemma 2 that, there exist at least $\chi(G)$ nodes in G, which are adjacent to $\chi(G) - 1$ nodes with degrees larger than $\chi(G) - 1$. Since η is the greatest number of nodes with a degree greater than or equal to η that are adjacent to at least $\eta - 1$ nodes, each of them with degree larger than or equal to $\eta - 1$, then $\chi(G) \leq \eta$. □

Remark 3. The proposed algorithm for computing η relies on the neighboring density. The neighboring density of node i is denoted by ρ_i and is defined as follows: ρ_i is the largest integer such that node i is adjacent to at least ρ_i nodes. Each of the latter has a degree greater than or equal to ρ_i. Algorithm 2 computes the neighboring density of all nodes. Then, η is computed by executing Algorithm 1, where d_i is replaced with ρ_i for all $i \in X$ and where ζ is replaced with η. The computational complexity for determining the neighboring density of all nodes is $\mathcal{O}(m\log_2(m))$, as it requires m operations to compute the degree, and $2m\log_2(2m)$ operations to sort $2m$ numbers (the degree sum of all nodes is $2m$). Therefore, the computational complexity for computing η is $\mathcal{O}(\max\{m\log_2(m), n\log_2(n)\})$.

Algorithm 2: Computing the neighboring density of all nodes.

```
Data: Graph G(X, U); where n ← |X| and m ← |U|.
Compute the degree of all nodes in X;
for i = 1 to n do
    Create the array tab by sorting the degree of the d_i neighbors of node i by non increasing order;
    ρ_i ← 0, stable ← 0, and j ← 0;
    while stable = 0 and j ≤ d_i do
        if tab[j] > ρ_i then
            ρ_i ← ρ_i + 1;
        else
            stable ← 1;
        j ← j + 1;
```

3. Theoretical quality assessment of these bounds

The three bounds introduced in this paper are compared theoretically to the five upper bounds from the literature, which were mentioned in the introduction, namely d, l, M, s and q.

Proposition 1. *For any simple, undirected, connected graph*

$$\xi \leq l.$$

Proof. The number of edges in any simple undirected graph is less than or equal to $n(n-1)/2$, thus:

$$\begin{aligned}
&2m \leq n^2 - n\\
&8m + 1 \leq 4n^2 - 4n + 1\\
&8m + 1 \leq (2n - 1)^2\\
&\sqrt{8m + 1} \leq 2n - 1\\
&1 - 2n \leq -\sqrt{8m + 1}\\
&4 - 8n \leq -4\sqrt{8m + 1}.
\end{aligned}$$

Then, $8m + 5$ is added to the last inequality

$$9 + 8(m - n) \leq (8m + 1) + 4 - 4\sqrt{8m + 1}$$
$$\sqrt{9 + 8(m - n)} \leq \sqrt{8m + 1} - 2$$
$$\frac{3 + \sqrt{9 + 8(m - n)}}{2} \leq \frac{1 + \sqrt{8m + 1}}{2}$$
$$\left\lfloor \frac{3 + \sqrt{9 + 8(m - n)}}{2} \right\rfloor \leq \left\lfloor \frac{1 + \sqrt{8m + 1}}{2} \right\rfloor$$
$$\xi \leq l. \quad \square$$

Proposition 2. *For any simple undirected graph*

$$\eta \leq \zeta.$$

Proof. This is obvious as the definition of ζ and η can be seen as the statement of two maximization problems. Since the requirements (or constraints) on η are more stringent than the requirements on ζ, the inequality $\eta \leq \zeta$ holds. $\square$

Proposition 3. *For any simple undirected graph*

$$\zeta \leq d.$$

Proof. Since $\delta(G)$ is the maximum degree in the graph, $d_v \leq \delta(G)$ for all $v \in X$. By definition of ζ, there exists at least one node w with a degree greater than or equal to $\zeta - 1$, then:

$$d_w \leq \delta(G)$$
$$\zeta - 1 \leq \delta(G)$$
$$\zeta \leq \delta(G) + 1$$
$$\zeta \leq d. \quad \square$$

Proposition 4. *For any simple undirected graph*

$$\zeta = M.$$

Proof. First, it is recalled that by definition of ζ, there does not exist $\zeta + 1$ nodes with a degree larger than or equal to ζ (otherwise this would be conflicting with the definition of ζ).

It is assumed without loss of generality that the nodes are indexed by non increasing degree: $d_1 \geq d_2 \geq \cdots \geq d_n$. Then it can be deduced that the nodes whose index is in $\{\zeta + 1, \ldots, n\}$ have a degree less than or equal to $\zeta - 1$.

The node set $X = \{1, \ldots, n\}$ is split into two subsets: $X = A \cup B$ with $A = \{1, \ldots, \zeta\}$ and $B = \{\zeta + 1, \ldots, n\}$. In other words, A is the set of the ζ nodes of highest degree, B is the set of the $n - \zeta$ nodes of lower degree.

For all i in X, we denote by m_i the minimum value between $d_i + 1$ and i (i.e. this makes it possible to write $M = \max_{i \in X} m_i$).

For all $i \in X$, i is either in A or in B:

- If $i \in A$, then node i is such that $d_i \geq \zeta - 1$, i.e. $d_i + 1 \geq \zeta$. Moreover, by definition of A, $i \leq \zeta$. Consequently:

  $$m_i = i \leq \zeta \leq d_i + 1 \quad \forall i \in A.$$

  In particular, for $i = \zeta$, $m_i = \zeta$, and by definition of M, $\zeta \leq M$.
- If $i \in B$, then node i is such that $d_i \leq \zeta - 1$, i.e. $d_i + 1 \leq \zeta$. Moreover, by definition of B, $i \geq \zeta$. Consequently:

  $$m_i = d_i + 1 \leq \zeta \leq i \quad \forall i \in B.$$

Finally, the inequality $m_i \leq \zeta$ holds for all $i \in \{1, \ldots, n\}$ and by definition of M this leads to $M \leq \zeta$. $\square$

Remark 4. Computing M by using the formula $M = \max_{i \in X} \min(d_i + 1, i)$ provided in [13] has a computational complexity of $\mathcal{O}(\max\{m, n \log_2 n\})$, as it requires computing the degree of the nodes, and sorting them by non increasing degree. Although ζ and M are defined differently, their computation requires the same order of arithmetic operations.

Proposition 5. *For any simple undirected graph*

$$\eta \leq s.$$

Proof. By definition of $\delta_2(G)$, there does not exist two adjacent nodes i and j in X such that $d_i > \delta_2(G)$ and $d_j > \delta_2(G)$. Consequently, it is impossible to find a node adjacent to at least $\delta_2(G) + 1$ nodes, whose degrees are at least $\delta_2(G) + 1$. This shows that $\eta - 1$ is less than or equal to $\delta_2(G)$, i.e. $\eta \leq s$. $\square$

Table 1
Upper bounds on the chromatic number.

Instances			Known upper bounds					New upper bounds		
Sour.	Name	$n\backslash m$	d	l	M	s	q	ξ	ζ	η
MYC	myciel3	11\20	6	6	5	4	6	6	5	4
MYC	myciel4	23\71	12	12	7	7	12	11	7	6
CAR	2-Insert._3	37\72	10	12	5	5	6	10	5	5
CAR	1-FullIns_3	30\100	12	14	9	12	12	13	9	7
CAR	3-Insert._3	56\110	12	15	5	5	7	12	5	5
MIZ	mug88_1	88\146	5	17	5	5	6	12	5	4
MIZ	mug88_25	88\146	5	17	5	5	6	12	5	4
CAR	4-Insert._3	79\156	14	18	5	5	8	14	5	5
SGB	queen5_5	25\160	17	18	13	13	17	18	13	13
MIZ	mug100_25	100\166	5	18	5	5	6	13	5	4
MIZ	mug100_1	100\166	5	18	5	5	6	13	5	4
CAR	2-FullIns_3	52\201	16	20	12	16	16	18	12	8
MYC	r125.1	125\209	9	20	7	7	10	11	7	6
CAR	1-Insert._4	67\232	23	22	9	9	16	19	9	7
MYC	myciel5	47\236	24	22	13	13	22	21	13	9
SGB	jean	80\254	37	23	12	14	19	20	12	11
SGB	queen6_6	36\290	20	24	16	16	20	24	16	16
SGB	huck	74\301	54	25	11	21	28	22	11	11
CAR	3-FullIns_3	80\346	20	26	14	20	20	24	14	10
SGB	miles250	128\387	17	28	13	15	16	23	13	10
SGB	david	87\406	83	29	16	31	42	26	16	12
SGB	queen7_7	49\476	25	31	21	19	25	30	21	19
SGB	anna	138\493	72	31	15	51	37	28	15	12
CAR	4-FullIns_3	114\541	24	33	16	24	24	30	16	12
CAR	2-Insert._4	149\541	38	33	9	11	20	29	9	9
CAR	1-FullIns_4	93\593	33	34	18	33	26	33	18	13
SGB	games120	120\638	14	36	13	14	15	33	13	11
SGB	queen8_8	64\728	28	38	24	22	28	37	24	22
DSJ	dsjc125.1	125\736	24	38	17	20	24	36	17	12
MYC	myciel6	95\755	48	39	21	25	44	37	21	14
CAR	5-FullIns_3	154\792	28	40	18	28	29	37	18	14
MYC	r250.1	250\867	14	42	13	13	15	36	13	10
CAR	3-Insert._4	281\1046	57	46	9	13	29	40	9	9
SGB	queen9_9	81\1056	33	46	27	25	33	45	27	25
SGB	miles500	128\1170	39	48	29	35	35	47	29	25
CAR	1-Insert._5	202\1227	68	50	17	24	46	46	17	13
SGB	queen8_12	96\1368	33	52	31	30	33	51	31	27
SGB	queen10_10	100\1470	36	54	32	28	36	53	32	28
CAR	2-FullIns_4	212\1621	56	57	24	56	51	54	24	16
SGB	homer	561\1628	100	57	25	56	51	47	25	18
CAR	4-Insert._4	475\1795	80	60	9	15	41	52	9	9
SGB	queen11_11	121\1980	41	63	35	31	41	62	35	31
SGB	miles750	128\2113	65	65	42	55	57	64	42	37
MYC	myciel7	191\2360	96	69	35	49	88	67	35	23
SGB	queen12_12	144\2596	44	72	38	34	44	71	38	34
SGB	miles1000	128\3216	87	80	57	82	74	80	57	49
DSJ	dsjc250.1	250\3218	39	80	33	35	39	78	33	25
CAR	1-FullIns_5	282\3247	96	81	36	96	73	78	36	23
SGB	queen13_13	169\3328	49	82	43	37	49	81	43	37
CAR	3-FullIns_4	405\3524	85	84	28	85	72	80	28	20
REG	zeroin_i3	206\3540	141	84	41	38	119	83	41	32
REG	zeroin_i2	211\3541	141	84	41	38	119	83	41	32
DSJ	dsjr500.1	500\3555	26	84	23	26	27	79	23	18
MYC	r125.5	125\3838	100	88	61	70	85	87	61	52
REG	mulsol_i2	188\3885	157	88	53	34	139	87	53	33
DSJ	dsjc125.5	125\3891	76	88	63	72	72	88	63	57
REG	mulsol_i3	184\3916	158	89	54	34	140	87	54	33
REG	mulsol_i1	197\3925	122	89	65	82	111	87	65	51
CAR	2-Insert._5	597\3936	150	89	20	39	76	83	20	17
REG	mulsol_i4	185\3946	159	89	54	34	140	88	54	33
REG	mulsol_i5	186\3973	160	89	55	34	141	88	55	33
REG	zeroin_i1	211\4100	112	91	54	95	104	89	54	51
HOS	ash331GPIA	662\4185	24	91	20	23	25	85	20	16
SGB	queen14_14	196\4186	52	92	46	40	52	90	46	40
SGB	queen15_15	225\5180	57	102	49	43	57	101	49	43

Table 1 (*continued*)

Instances			Known upper bounds					New upper bounds		
Sour.	Name	$n\backslash m$	d	l	M	s	q	ξ	ζ	η
SGB	miles1500	128\5198	107	102	84	106	96	102	84	78
LEI	le450_5a	450\5714	43	107	34	35	44	104	34	25
LEI	le450_5b	450\5734	43	107	34	35	43	104	34	26
SGB	queen16_16	256\6320	60	112	54	46	60	111	54	46
CAR	1-Insert._6	607\6337	203	113	33	69	136	108	33	25
CAR	4-FullIns_4	690\6650	120	115	36	120	104	110	36	24
DSJ	dsjc125.9	125\6961	121	118	109	113	116	118	109	106
HOS	will199GPIA	701\7065	42	119	35	35	42	114	35	28
MYC	r125.1c	125\7501	125	122	116	116	123	122	116	116
HOS	ash608GPIA	1216\7844	21	125	20	20	22	116	20	16
LEI	le450_15a	450\8168	100	128	57	68	93	125	57	39
LEI	le450_15b	450\8169	95	128	56	72	88	125	56	39
LEI	le450_25a	450\8260	129	129	63	85	114	126	63	46
LEI	le450_25b	450\8263	112	129	60	80	99	126	60	43
REG	fpsol2i3	425\8688	347	132	53	68	299	130	53	35
REG	fpsol2i2	451\8691	347	132	53	68	299	129	53	35
CAR	3-Insert._5	1406\9695	282	139	25	58	142	130	25	17
LEI	le450_5d	450\9757	69	140	52	53	68	137	52	41
LEI	le450_5c	450\9803	67	140	52	55	67	138	52	41
CAR	5-FullIns_4	*1085\11 395*	161	151	49	161	142	145	49	28
REG	fpsol2i1	496\11 654	253	153	79	102	231	150	79	67
CAR	2-FullIns_5	852\12 201	216	156	56	216	193	152	56	31
DSJ	dsjc500.1	500\12 458	69	158	59	61	69	156	59	47
HOS	ash958GPIA	*1916\12 506*	25	158	21	22	26	147	21	17
REG	inithx_i3	621\13 969	543	167	52	235	476	164	52	38
REG	inithx_i2	645\13 979	542	167	52	235	476	164	52	38
MYC	r1000.1	*1000\14 378*	50	170	41	47	51	165	41	34
SCH	school1_nsh	352\14 612	233	171	101	115	195	170	101	84
MYC	r250.5	250\14 849	192	172	119	154	166	172	119	99
DSJ	dsjc250.5	250\15 668	148	177	126	134	141	177	126	116
LEI	le450_15c	450\16 680	140	183	93	129	133	181	93	70
LEI	le450_15d	450\16 750	139	183	92	129	131	182	92	70
LEI	le450_25c	450\17 343	180	186	101	128	163	185	101	76
LEI	le450_25d	450\17 425	158	187	99	138	145	185	99	75
REG	inithx_i1	864\18 707	503	193	074	239	441	190	74	57
SCH	school1	385\19 095	283	195	117	172	213	194	117	98
CUL	flat300_20_0	300\21 375	161	207	144	148	155	206	144	135
CUL	flat300_26_0	300\21 633	159	208	146	152	154	208	146	136
CUL	flat300_28_0	300\21 695	163	208	146	157	158	208	146	136
GOM	qg.order30	900\26 100	59	228	59	59	60	226	59	59
DSJ	dsjc250.9	250\27 897	235	236	219	224	228	236	219	214
MYC	r250.1c	250\30 227	250	246	238	242	246	246	238	236
CAR	3-FullIns_5	*2030\33 751*	410	260	79	410	343	253	79	40
KOS	wap05a	905\43 081	229	294	147	200	213	291	147	106
KOS	wap06a	947\43 571	231	295	147	200	211	293	147	105
DSJ	dsjc1000.1	*1000\49 629*	128	315	112	112	127	313	112	93
DSJ	dsjr500.5	500\58 862	389	343	234	282	347	343	234	197
GOM	qg.order40	*1600\62 400*	79	353	79	79	80	350	79	79
DSJ	dsjc500.5	*500\62 624*	287	354	251	260	277	353	251	236
HOS	abb313GPIA	*1557\65 390*	188	362	123	119	184	358	123	94
CAR	4-FullIns_5	*4146\77 305*	696	393	96	696	598	384	96	48
KOS	wap07a	*1809\103 368*	299	455	188	259	275	452	188	130
KOS	wap08a	*1870\104 176*	309	456	189	272	293	453	189	129
KOS	wap01a	*2368\110 871*	289	471	174	223	270	467	174	115
KOS	wap02a	*2464\111 742*	295	473	175	222	280	469	175	116
DSJ	dsjc500.9	*500\112 437*	472	474	443	450	461	474	443	437
DSJ	dsjr500.1c	*500\121 275*	498	492	478	489	490	492	478	476
GOM	qg.order60	*3600\212 400*	119	652	119	119	120	647	119	119
MYC	r1000.5	*1000\238 267*	782	690	472	535	696	690	472	396
CUL	flat1000_50	*1000\245 000*	521	700	492	503	511	700	492	474
CUL	flat1000_60	*1000\245 830*	525	701	493	501	515	701	493	472
CUL	flat1000_76	*1000\246 708*	533	702	494	501	523	702	494	474
DSJ	dsjc1000.5	*1000\249 826*	552	707	501	518	538	706	501	475
KOS	wap03a	*4730\286 722*	345	757	230	302	333	752	230	148
KOS	wap04a	*5231\294 902*	352	768	238	307	341	762	238	149

(*continued on next page*)

Table 1 (*continued*)

Instances			Known upper bounds					New upper bounds		
Sour.	Name	$n\backslash m$	d	l	M	s	q	ξ	ζ	η
LAT	latinsquare10	*900\307 350*	684	784	684	684	685	784	684	684
DSJ	dsjc1000.9	*1000\449 449*	925	948	888	912	910	948	888	877
MYC	r1000.1c	*1000\485 090*	992	985	957	976	978	985	957	951
GOM	qg.order100	*10 000\990 000*	199	1407	199	199	200	1401	199	199
MYC	c2000.5	*2000\999 836*	1075	1414	1000	1028	1054	1414	1000	962
MYC	c4000.5	*4000\4 000 268*	2124	2829	2002	2019	2093	2828	2002	1942
Average number of colors			186.1	218.5	122.2	147.5	171.2	215.9	122.2	108.9
Av. improvement of η (in%)			−46.1	−58.3	−18.4	−29.4	−42.8	−56.6	−18.4	0.0
Total time (s)			0.2	0.0	0.3	2.1	0.3	3.8	0.9	14.0

Table 2
Computational assessment of Propositions 1–6 based on Table 1.

Propositions		Avg. improvement (%)
Proposition 1	$\xi \leq l$	−4.56
Proposition 2	$\eta \leq \zeta$	−18.36
Proposition 3	$\zeta \leq d$	−35.99
Proposition 4	$\zeta = M$	0.00
Proposition 5	$\eta \leq s$	−29.39
Proposition 6	$\zeta \leq q$	−32.02

Proposition 6. *For any simple undirected graph*

$$\zeta \leq q.$$

Proof. We prove by contradiction that $\zeta \leq q$ by using Proposition 4.

$$\zeta = M = \max_{i \in X} \min(d_i + 1, i).$$

We denote by A and B the two subsets of X: $A = \{1, \ldots, \zeta\}$ and $B = \{\zeta + 1, \ldots, n\}$.
As shown in the proof of Proposition 4:

$$i \leq \zeta \leq d_i + 1 \quad \forall i \in A$$
$$d_i + 1 \leq \zeta \leq i \quad \forall i \in B.$$

We assume that $\zeta > q$.
First, it is recalled that Stacho has proved in [12] that $d_q < q$, i.e. $d_q + 1 \leq q$. Then $\zeta > q$ does not hold if $q \in A$.
Second, if q belongs to B it must satisfy $\zeta \leq q$ which conflicts with the hypothesis $\zeta > q$.
Consequently, this proves that $\zeta \leq q$. □

4. Computational assessment of these bounds

The new bounds introduced in this paper are compared to the five bounds of the literature on the DIMACS instances [5] for graph coloring. The detailed results are shown in Table 1. The first three columns of this table provide the instance source at DIMACS, its name, the number of nodes and the number of edges. The next eight columns show the upper bound on the number of colors provided by the five bounds of the literature, and the three upper bounds introduced in this paper. The last three rows of Table 1 show the average value of each bound on the DIMACS instances, the penultimate row is the average improvement provided by η over all the other bounds (note that these figures are not computed on the average numbers of colors), and the last row is the total amount of CPU time (in seconds) required for computing each bound on an Intel x eon processor system at 2.67 GH z and 8 G bytes RAM. Algorithms have been implemented in c ++ and compiled with gcc 4.11 on a Linux System.

Table 2 is displayed to assess the practical strength of Propositions 1–6. As each proposition is of the form $a \leq b$ (except Proposition 4), the last column of Table 2 indicates by which amount bound a is better than bound b (the average improvement is defined as the average value of $(a - b)/b$ over all the instances, in percent). Naturally, this amount is 0% in the particular case of Proposition 4, as it is an equality. It can be seen that ξ does not provide a significant advantage over l in practice.

However, Propositions 2, 3, 5 and 6 are stronger as the improvement is larger than 18%. More specifically, the best bound proposed in this paper outperforms the best upper bound of the literature by more than 18% in average. Proving that $M = \zeta$ is important for highlighting the reason for the practical superiority of η over M. Indeed, η is based on the same principle as ζ; it focuses on the degrees of saturation of nodes. The difference is that, η goes one step further than ζ by considering

the degree of saturation of the neighbors of each nodes (i.e. the so-called neighboring density). This additional requirement has a computational cost which is drastically larger than the one required by computing ζ, but it provides a significant improvement in terms of the upper bound quality.

5. Conclusion

This paper introduces three new upper bounds on the chromatic number, without making any assumptions on the graph structure. The first one, ξ, is based on the number of edges and nodes and only requires connectivity, whereas ζ and η are based on the degree of the nodes in the graph. It is shown that ζ is equal to an existing bound, while being computed in a very different way. Moreover, a series of inequalities are proved, showing that these new bounds outperform five of the most well-known upper bounds from the literature. Computational experiments also show that the best bound proposed in this paper, η, is significantly better than the five bounds of the literature, and highlight the benefit of using the degree of saturation and its refined version (the neighboring density) for producing competitive upper bounds for graph coloring.

References

[1] D. Brélaz, New methods to color the vertices of a graph, Communications of the Association for Computing Machinery 22 (1979) 251–256.
[2] T. Bui, T. Nguyen, C. Patel, K. Phan, An ant-based algorithm for coloring graphs, Discrete Applied Mathematics 156 (2) (2008) 190–200.
[3] M. Caramia, P. Dell'Olmo, Coloring graphs by iterated local search traversing feasible and infeasible solutions, Discrete Applied Mathematics 156 (2) (2008) 201–217.
[4] R. Diestel, Graph Theory, in: Ser. Graduate Texts in Mathematics, vol. 173, Springer-Verlag, Heidelberg, 2005, [Online]. Available: http://diestel-graph-theory.com/GrTh.html.
[5] Dimacs, 2011 [Online]. Available: http://mat.gsia.cmu.edu/COLOR/instances.html.
[6] Graph coloring, 2009 [Online]. Available: http://en.wikipedia.org/wiki/Graph/_coloring.
[7] W. Klotz, Graph coloring algorithms, Mathematik-Bericht, TU Clausthal, 2002, pp. 1–9.
[8] D.E. Knuth, The sandwich theorem, The Electronic Journal of Combinatorics 1 (1) (1994) 1–49.
[9] A. Mehrotra, M. Trick, A column generation approach for graph coloring, INFORMS Journal on Computing 8 (4) (1996) 344–354.
[10] I. Méndez-Díaz, P. Zabala, A cutting plane algorithm for graph coloring, Discrete Applied Mathematics 156 (2) (2008) 159–179.
[11] L. Stacho, New upper bounds for the chromatic number of a graph, Journal of Graph Theory 36 (2) (2001) 117–120.
[12] L. Stacho, A note on upper bound for the chromatic number of a graph, Acta Mathematica Universitatis Comenianae 71 (1) (2002) 1–2.
[13] D. Welsh, M. Powell, An upper bound for the chromatic number of a graph and its application to timetabling problems, The Computer Journal 10 (1) (1967) 85–86.